ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΠΛΗΡΟΦΟΡΙΚΗ ΚΑΙ ΥΠΟΛΟΓΙΣΤΙΚΗ ΒΙΟΙΑΤΡΙΚΗ

Συμβολή Αλγορίθμων Δρομολόγησης στη Βελτιστοποίηση Αρχιτεκτονικών Noc

Αυγέρης Αθανάσιος

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
Επιβλέπων
Λαλλάς Ευθύμιος

Λαμία, 2017

UNIVERCITY OF THESSALY

SCHOOL OF SCIENCE

INFORMATICS AND COMPUTATIONAL BIOMEDICINE

Contribution of Routing Algorithms to Optimization NoC Architectures

Avgeris Athanasios

Master thesis

Lallas Eythymios

Lamia 2017

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΔΙΑΤΜΗΜΑΤΙΚΟ ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΠΛΗΡΟΦΟΡΙΚΗ ΚΑΙ ΥΠΟΛΟΓΙΣΤΙΚΗ ΒΙΟΙΑΤΡΙΚΗ

«ΠΛΗΡΟΦΟΡΙΚΗ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗΝ ΑΣΦΑΛΕΙΑ, ΔΙΑΧΕΙΡΙΣΗ ΜΕΓΑΛΟΥ ΟΓΚΟΥ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ»

Συμβολή Αλγορίθμων Δρομολόγησης στη Βελτιστοποίηση Αρχιτεκτονικών Noc

Αυγέρης Αθανάσιος

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων
Λαλλάς Ευθύμιος

Λαμία, 2017

«Υπεύθυνη Δήλωση μη λογοκλοπής και ανάληψης προσωπικής ευθύνης»

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, και γνωρίζοντας τις συνέπειες της λογοκλοπής, δηλώνω υπεύθυνα και ενυπογράφως ότι η παρούσα εργασία με τίτλο [«Αλγόριθμοι δρομολόγησης σε αρχιτεκτονικές NoC»] αποτελεί προϊόν αυστηρά προσωπικής εργασίας και όλες οι πηγές από τις οποίες χρησιμοποίησα δεδομένα, ιδέες, φράσεις, προτάσεις ή λέξεις, είτε επακριβώς (όπως υπάρχουν στο πρωτότυπο ή μεταφρασμένες) είτε με παράφραση, έχουν δηλωθεί κατάλληλα και ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο ΔΗΛΩΝ

Ημερομηνία

Υπογραφή

Συμβολή Αλγορίθμων Δρομολόγησης στη Βελτιστοποίηση Αρχιτεκτονικών Noc

Αυγέρης Αθανάσιος

Τριμελής Επιτροπή:

Λαλλάς Ευθύμιος, Επίκουρος Καθηγητής (Τ.Ε.Ι Στερεάς Ελλάδας)

Σταμούλης Γεώργιος, Καθηγητής (Πανεπιστημίου Θεσσαλίας)

Λουκόπουλος Αθανάσιος, Επίκουρος Καθηγητής (Πανεπιστημίου Θεσσαλίας)

Επιστημονικός Σύμβουλος:

Λαλλάς Ευθύμιος, Επίκουρος Καθηγητής (Τ.Ε.Ι Στερεάς Ελλάδας)

ΕΥΧΑΡΙΣΤΙΕΣ

Η ολοκλήρωση αυτής της πτυχιακής υλοποιήθηκε με την υποστήριξη ενός αριθμού ανθρώπων στους οποίους θα ήθελα να εκφράσω τις θερμότερες ευχαριστίες μου. Πρώτα από όλους θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Λαλλά Ευθύμιο για την άψογη συνεργασία μας και τις πολύτιμες συμβουλές του. Επίσης θα ήθελα να ευχαριστήσω τη γυναίκα μου και το υιό μου για την κατανόηση και την υποστήριξή τους στην απόφασή μου να ξανακαθίσω στα φοιτητικά έδρανα για την απόκτηση ενός ακόμα πτυχίου.

Αυγέρης Β. Αθανάσιος
Οκτώβριος 2017

ΠΕΡΙΛΗΨΗ

Στο σύγχρονο κόσμο οι απατήσεις για ολοένα μεγαλύτερες ταχύτητες μεταφοράς δεδομένων είναι κάτι παραπάνω από επιβεβλημένες. Τα δίκτυα σε ψηφίδα ή NoC (Network on Chip) αποτελούν μια από τις πιο ενδεδειγμένες λύσεις σε αυτή την κατεύθυνση. Ο αναγνώστης της παρούσας εργασίας έχει τη δυνατότητα να κατανοήσει βασικές έννοιες που διέπει ένα NoC, τις αρχιτεκτονικές σύνδεσης τους και τους κανόνες που το διέπουν. Παρουσιάζονται επίσης βασικές και προχωρημένες τεχνικές δρομολόγησης των πακέτων, και συγκριτικές προσομοιώσεις αλγορίθμων δρομολόγησης όσον αφορά την ταχύτητα, την αξιοπιστία, την κατανάλωση ενέργειας και την περιοχή που καταλαμβάνουν τα συστήματα αυτά.

ΠΕΡΙΕΧΟΜΕΝΑ

$1^{ο}$ ΚΕΦΑΛΑΙΟ

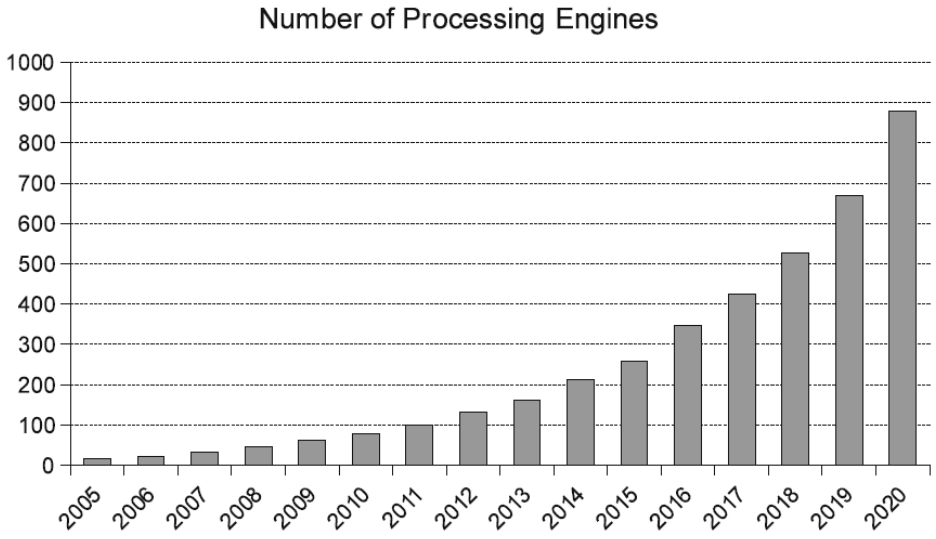
ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ ΣΤΑ NETWORK ON CHIP

1.1 Εισαγωγή

Δεδομένου ότι ο αριθμός των πυρήνων που ενσωματώνονται στα συστήματα NoC (Network on Chip) αυξάνετε συνεχώς, ο ρόλος που διαδραματίζουν στα συστήματα επικοινωνίας γίνετε ολοένα και πιο σημαντικός. Στη σημερινή εποχή τα Noc χαρακτηρίζονται ως η πιο "βιώσιμη" λύση στο χώρο των επικοινωνιών. Σε αυτό το κεφάλαιο θα αναλύσουμε βασικές έννοιες πάνω στα NoC συστήματα, συμπεριλαμβανομένων τοπολογιών δικτύου, τεχνικές μεταγωγής δεδομένων και αλγορίθμων δρομολόγησης.

Προγραμματιζόμενα chip, όπως οι μικροεπεξεργαστές αναπτύχθηκαν αρχικά με έναν μόνο υπολογιστικό πυρήνα. Μπορούσαν να εκτελέσουν τα προγράμματα εντολή προς εντολή, με ταχύτητα ανάλογη της συχνότητας χρονισμού τους. Έτσι η αύξηση της ταχύτητας του ρολογιού ήταν και ο πιο συνηθισμένος τρόπος στη βελτίωση της συνολικής απόδοσης του chip. Δυστυχώς η αύξηση της συχνότητας χρονισμού, έχει πολλά μειονεκτήματα. Το βασικότερο είναι πως μια τέτοια αύξηση είναι άρρηκτα συνδεδεμένη τόσο με την αύξηση της ισχύος, όσο κυρίως με την αύξηση της εκλυόμενης θερμότητας από το σύστημα. Αυτό έχει σαν αποτέλεσμα την αύξηση της

θερμοκρασίας, η οποία δυστυχώς περιορίζεται σε μια πολύ μικρή επιφάνεια, μόλις λίγων τετραγωνικών εκατοστών. Η αύξηση της θερμοκρασίας λειτουργίας επηρεάζει επίσης και την αξιοπιστία του συστήματος, καθώς μια αύξηση 10^0 C συνεπάγετε και διπλασιασμό τις πιθανότητας αστοχίας του υλικού μας.



Εικόνα 1.1: Αναμενόμενος αριθμός στοιχείων σε NoC

Όπως είναι φυσικό η ολοένα αυξανόμενη ζήτηση σε επεξεργαστική ισχύ σε συνδυασμό με τις εφαρμογές "νέας γενιάς" δεν θα μπορούσε απλά να προέρθει μόνο από την αύξηση της συχνότητας του ρολογιού. Για να ξεπεράσουμε τους τεχνικούς περιορισμούς στην αύξηση της ταχύτητας του ρολογιού, τα σύγχρονα chip είναι κατασκευασμένα με πολλούς πυρήνες οι οποίοι μπορούν να επεξεργαστούν τα δεδομένα παράλληλα, αλλά με μικρότερη ταχύτητα χρονισμού.

Καθώς ο αριθμός των πυρήνων αυξάνετε συνεχώς (εικόνα 1.1) τα θέματα που σχετίζονται με τις on-chip επικοινωνίες γίνονται ολοένα και πιο σημαντικά έναντι των υπολογιστικών θεμάτων. Δηλαδή, μπορούμε να έχουμε την επεξεργαστική ισχύ που χρειαζόμαστε (αυξάνοντας π.χ. τον αριθμό των

πυρήνων), αλλά την ίδια στιγμή υπάρχει η ανάγκη ενός αποτελεσματικότερου δικτύου επικοινωνιών για να γεφυρωθεί το μεταξύ τους χάσμα.

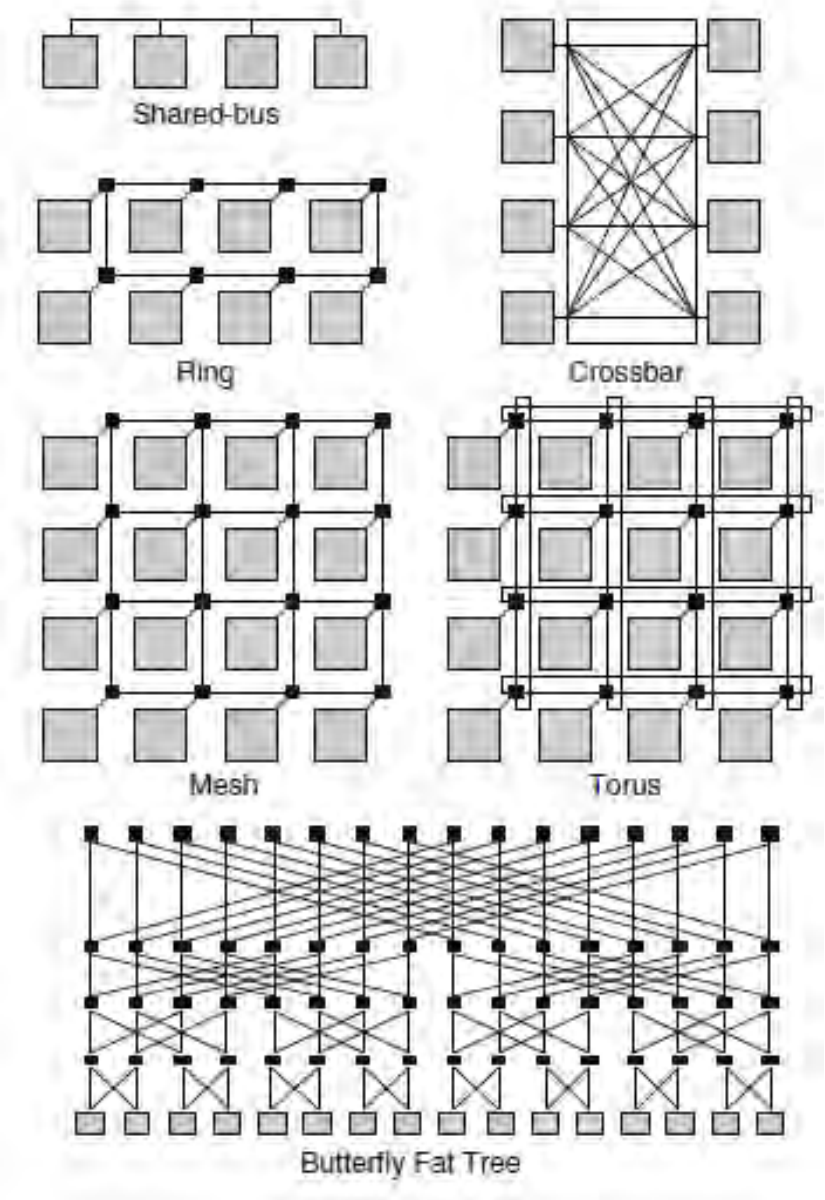
Τα Network-on-Chip (NoC) θεωρούνται σήμερα ως η πλέον βιώσιμη λύση σε θέματα επικοινωνιών σε συνάρτηση με την επόμενη γενιά πολλαπλών πυρήνων System-on-Chip (SoC). Το NoC μπορεί να το φανταστεί κανείς σαν ένα δίκτυο υπολογιστών, όπου οι υπολογιστές έχουν αντικατασταθεί με chip.

Τα NoCs έχουν χαρακτηριστεί ευέλικτα, αξιόπιστα και ιδανικά αλληλένδετα με τα MPSoCs [1], [2], [3] . Ωστόσο το οι πόροι που καταναλώνονται από το δίκτυο καταλαμβάνουν ένα σημαντικό μέρος της ισχύς του συστήματος [4]. Δεδομένου ότι η κατανάλωση ενέργειας είναι από τα σημαντικότερα ζητήματα στα σημερινά chip των δισεκατομμυρίων τρανζίστορ, επιπλέον της καθυστέρησης και της περιοχής του δικτύου, θα πρέπει να λάβουμε σοβαρά υπόψη μας και την κατανάλωση κατά τη διασύνδεση των επιμέρους στοιχείων του συστήματός μας. Ως εκ τούτου, ένα NoC θα πρέπει να συμπεριληφθεί σε μια συγκεκριμένη περιοχή πυριτίου με τη την αποδοτικότερη τοπολογία και το βέλτιστο αλγόριθμο δρομολόγησης.

1.2 Τοπολογίες Δικτύου

Σαν συνδεσμολογία δικτύου χαρακτηρίζουμε τη μελέτη των ρυθμίσεων και της συνδεσμολογίας των δρομολογητών. Με άλλα λόγια ρυθμίζουμε τα διαθέσιμα κανάλια και τα μοτίβα σύνδεσης που είναι διαθέσιμα για τη μεταφορά δεδομένων μέσω δικτύου. Απόδοση, κόστος και επεκτασιμότητα είναι οι σημαντικότεροι παράγοντες για την επιλογή της κατάλληλης τοπολογίας. Shared-Bus, Crossbar, Butterfly Fat-Tree, Ring, Torus και 2D-Mesh είναι οι δημοφιλέστερες τοπολογίες για τις τεχνολογίες on-chip [5], [6].

Τα ενσύρματα δίκτυα έχουν τουλάχιστον έναν επεξεργαστή (Processing Element) σε κάθε δρομολογητή, έτσι ώστε μέσω των δρομολογητών αν μπορούμε να εξαπλώσουμε το δίκτυ μεταξύ των PE's. Κάτι τέτοιο βοηθά στο να απλοποιηθεί η φυσική υλοποίηση ενός δικτύου. Shared-bus, ring, and 2D mesh/torus topologies (εικόνα 1.2) είναι παραδείγματα τοπολογιών άμεσων δικτύων τα οποία ναι μεν παρέχουν τεράστια βελτίωση στην απόδοση, αλλά το κόστος υλικού αυξάνεται δραματικά από την εκθετική αύξηση του αριθμού των PEs. Από την άλλη, τα έμμεσα δίκτυα έχουν ένα σύνολο δρομολογητών οι οποίοι δεν είναι συνδεδεμένοι σε κάποιο PE. Σε όλες τις δενδροειδής τοπολογίες, τα PE συνδέονται μόνο σε δρομολογητές φύλλων της διάταξης (π.χ. τοπολογία butterfly) καθώς και ο ραβδεπαφικός μεταγωγέας (crossbar switch) θεωρούνται έμμεσα δίκτυα (εικόνα 1.2).



Εικόνα 1.2 Τοπολογίες δικτύων Shared-bus, Ring, Crossbar, Mesh, Torus, and Butterfly

Η τοπολογία shared-bus είναι η απλούστερη όλων και χρησιμοποιεί μόνο μία κοινή σύνδεση μεταξύ όλων των PE όπου ανταγωνίζονται για την αποκλειστική πρόσβαση στο δίαυλο. Για την επικοινωνία οι εφαρμογές πρέπει να ξεπεράσουν τους περιορισμούς του εύρους ζώνης της τοπολογίας shared-bus και να προχωρήσουμε σε κλιμακούμενα δίκτυα. Ωστόσο οι

τοπολογίες παίζουν ολοένα και μικρότερο ρόλο καθώς ο αριθμός των PE στα δίκτυα αυξάνετε συνεχώς. Μια μικρή τροποποίηση στην τοπολογία shared-bus επιτρέπει περισσότερες ταυτόχρονες μεταγωγές πακέτων δημιουργώντας την τοπολογία δακτυλιδιού όπου κάθε PE έχει ακριβώς δύο γείτονες. Σε αυτή την τοπολογία τα μηνύματα μεταφέρονται μεταξύ των PE μέχρι να φτάσουν στον τελικό προορισμό τους. Η τοπολογία crossbar επιτρέπει κάθε PE της τοπολογίας να είναι άμεσα συνδεδεμένο με τα υπόλοιπα PE.

Οι δενδροειδείς τοπολογίες πάσχουν στο γεγονός ότι ο αριθμός των δρομολογητών υπερβαίνει τον αριθμό των PE, όσο αυξάνετε ο αριθμός των PE. Αυτό επισύρει μια σημαντική επιφόρτιση του δικτύου. Για τις on-chip διασυνδέσεις δικτύου, αυτό είναι πολύ πιο σημαντικό από ότι για τις off-chip διασυνδέσεις και η επεκτασιμότητα του δικτύου είναι επίσης πολύ σημαντική. Τα δίκτυα βασισμένα στις τοπολογίες Mesh και Torus χρησιμοποιούνται ευρέως στις αρχιτεκτονικές πολλαπλών πυρήνων λόγω της απλής διασύνδεσή τους και της εύκολης δρομολόγησης μεταξύ γειτονικών πυρήνων. Οι συγκεκριμένες τοπολογίες είναι πλήρως επεκτάσιμες. Παρά το γεγονός ότι η τοπολογία Torus καλύτερη απόδοση η κανονικότητα, η καλύτερη αξιοποίηση διασυνδέσεων είναι πλεονεκτήματα της τοπολογίας Mesh. Με άλλα λόγια η τοπολογία Mesh χρησιμοποιεί λιγότερους δρομολογητές. Εν ολίγης κάθε τοπολογία έχει τα πλεονεκτήματα και τα μειονεκτήματά της.

1.3 Μηχανισμοί Μεταγωγής

Οι μηχανισμοί μεταγωγής καθορίζουν τη διαδρομή που θα ακολουθήσουν τα μηνύματα μέσα σε ένα δίκτυο. Ο στόχος είναι να διαμοιράζονται οι πόροι του δικτύου μας μεταξύ των μηνυμάτων που διέρχονται από αυτό. Βασικά, τα κυκλώματα μεταγωγής και η μεταγωγή ενός πακέτου αποτελούν τα δύο άκρα του μηχανισμού μεταγωγής.

Στη μεταγωγή κυκλώματος, δημιουργείται ένα κύκλωμα από μια πηγή στον προορισμό πριν ξεκινήσει η μετάδοση των δεδομένων και παραμένει μέχρι να μεταδοθεί το μήνυμα στον προορισμό του (όπως ακριβώς συμβαίνει και στα ενσύρματα τηλεφωνικά δίκτυα όπου κατά την πραγματοποίηση μια κλίσης δημιουργείτε ένα κύκλωμα το οποίο παραμένει ανοιχτό μέχρι την αποπεράτωσή της). Ο μηχανισμός αυτός έχει μικρή καθυστέρηση και εγγυάται το εύρος ζώνης αλλά, πάσχει από το γεγονός ότι δεσμεύει τα κανάλια για μεγάλο χρονικό διάστημα, έχει χαμηλή απόδοση και απαιτεί μεγάλο χρονικό διάστημα για την εγκατάσταση μιας σύνδεσης.

Η μεταγωγή πακέτων είναι ένας εναλλακτικός μηχανισμός όπου τα δεδομένα δεν μεταβιβάζονται σε προκαθορισμένα κυκλώματα. Ένα μήνυμα μπορεί να διαιρεθεί σε άλλα μικρότερα τα οποία μοιράζονται στα κανάλια του κυκλώματος τα οποία μπορούν ταυτόχρονα να χρησιμοποιούνται και από άλλα πακέτα. Κάθε πακέτο αποτελείται από την επικεφαλίδα που περιλαμβάνει πληροφορίες για τη δρομολόγηση του πακέτου, ελέγχου κ.α., από τα προς μετάδοση δεδομένα και πιθανώς από μια ουρά. Τα δεδομένα ακολουθούν τη διαδρομή που έχει καθοριστεί στην επικεφαλίδα, ενώ η ουρά αποδεσμεύει το κύκλωμα. Τα πακέτα μεμονωμένα ή ανεξάρτητα δρομολογούνται μέσω δικτύου και στον προορισμό συγκεντρώνονται και επαναδημιουργούν το αρχικό μας πακέτο. Αν ένα μήνυμα έχει χωριστεί σε μικρότερα πακέτα, θα πρέπει η σειρά άφιξής τους στο PE να είναι η ίδια με τη σειρά αναχώρησης. Ως εκ τούτου, η ακρίβεια στην παράδοση των πακέτων μας είναι από τις βασικές προτεραιότητες ενός on-chip δικτύου. Ο μηχανισμός μεταγωγής πακέτου αξιοποιεί καλύτερα ένα κανάλι βελτιώνει την διεκπεραιωτικότητα του.

Στη μεταγωγή πακέτων, ο έλεγχος ροής του buffer καθορίζει το μηχανισμό που ασχολείται τόσο με την κατανομή των καναλιών όσο και του συνολικού διαθέσιμου buffer κατά την μεταφορά των πακέτων από την πηγή στον

προορισμό τους. Ο μηχανισμός ελέγχου ροής είναι αναγκαίος όταν δύο ή περισσότερα πακέτα προσπαθήσουν να χρησιμοποιήσουν το ίδιο κανάλι την ίδια χρονική στιγμή. Συνήθως χρησιμοποιούνται τρεις διαφορετικές στρατηγικές ελέγχου ροής: η Virtual Cut-Through, η Virtual Cut-Through και η wormhole. Όταν αυτοί οι μηχανισμοί εφαρμοστούν σε on-chip δίκτυα έχουν διαφορετικές αποδόσεις και διαφορετικές απαιτήσεις σε πόρους δικτύου.

1.3.1 Αποθήκευση-και-προώθηση (Store-and-Forward)

Είναι ο απλούστερος μηχανισμός ελέγχου ροής του δικτύου μας. Στην προσέγγιση που ακολουθεί, κάθε δρομολογητής κατά μήκος της διαδρομής αποθηκεύει ολόκληρο το μήνυμα στο buffer, και στη συνέχεια το πακέτο προωθείτε σε επιλεγμένους γειτονικούς δρομολογητές, αν αυτοί διαθέτουν τον απαραίτητο χώρο στο buffer τους ώστε να φιλοξενήσουν ολόκληρο το πακέτο. Ο μηχανισμός αυτός προϋποθέτει μεγάλο buffer (τουλάχιστον ίσο με αυτό του πακέτου) σε κάθε δρομολογητή του δικτύου, το οποίο θα μπορούσε να αυξήσει το κόστος δραματικά. Επιπροσθέτως θα καθυστερούσε σημαντικά τη διάδοση ενός πακέτου αφού αυτό δεν μπορούσε να μεταβιβαστεί στον επόμενο δρομολογητή, αν δε είχε φορτωθεί πλήρως από τον προηγούμενο. Συμπερασματικά η εν λόγο προσέγγιση δεν θεωρείτε πρακτική σε μεγάλης κλίμακας NoCs.

1.3.2 Εικονική Εισχώρηση (Virtual Cut-Through)

Ο συγκεκριμένος μηχανισμός (Virtual Cut-Through) [5] κατασκευάστηκε για την αντιμετώπιση της λανθάνουσας καθυστέρησης στα μεγάλα δίκτυα που χρησιμοποιούν τον μηχανισμό Store-and-Forward μειώνοντας ουσιαστικά την καθυστέρηση των πακέτων σε κάθε στάδιο δρομολόγησης. Σε αυτή την προσέγγιση, ένα πακέτο μπορεί να προωθηθεί στον επόμενο δρομολογητή πριν την ολοκλήρωση της λήψης του από τον επόμενο, μειώνοντας έτσι την

καθυστέρηση της (Store-and-Forward) προσέγγισης. Ωστόσο, όταν ο δρομολογητής του επόμενου σταδίου δεν είναι διαθέσιμος, ακόμα και αυτή η διαδικασία απαιτεί μεγαλύτερο χώρο στο buffer του κάθε δρομολογητή, ώστε αυτός να είναι σε θέση να αποθηκεύσει όλο το πακέτο.

1.3.3 Μηχανισμός "Σκουληκότρυπα" (Wormhole)

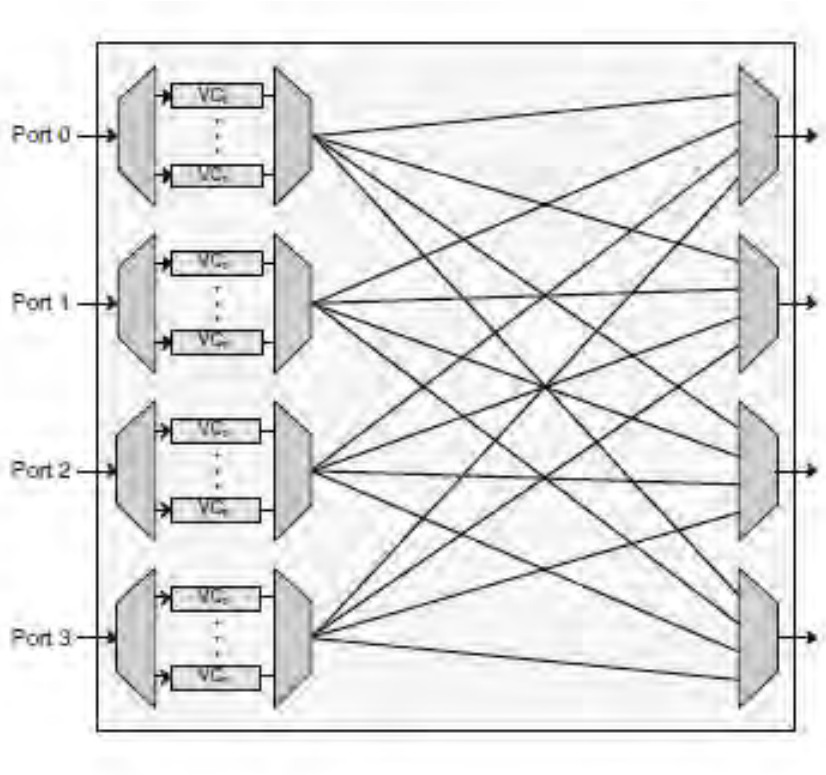
Σε αυτόν τον μηχανισμό τα πακέτα σπάνε σε μικρότερα τμήματα που ονομάζονται FLITs (FLow control digIT) [7]. Στη συνέχεια τα flits, δρομολογούνται στο δίκτυο το ένα μετά το άλλο, μέσω της διαδικασίας pipeline. Το πρώτο flit του κάθε πακέτου (header) δεσμεύει την διαδρομή που θα ακολουθήσουν τα υπόλοιπα flits, το κύριο σώμα των flits (payload) ακολουθούν την προδιαγεγραμμένη διαδρομή του πρώτου και το τελευταίο flit αποδεσμεύει την διαδρομή. Ο μηχανισμός wormhole δεν απαιτεί από το σύστημα την αποθήκευση όλου του πακέτου στο δρομολογητή καθώς περιμένει την κεφαλίδα ενός flit να μεταβεί σε κάποιο από τα επόμενα στάδια. Έτσι ένα πακέτο μπορεί να καταλάβει ταυτόχρονα πολλούς ενδιάμεσους δρομολογητές. Με άλλα λόγια η προσέγγιση wormhole είναι παρόμοια με την προσέγγιση Virtual Cut-Through αλλά εδώ η κατανομή των καναλιών και του buffer γίνετε με βάση τα flit και όχι τα πακέτα. Κατά συνέπεια η προσέγγιση wormhole απαιτεί λιγότερο χώρο στο buffer και επιτρέπει τη γρήγορη διάδοση των πακέτων μέσα από μικρότερους δρομολογητές. Λόγω αυτών των πλεονεκτημάτων ο συγκεκριμένος μηχανισμός είναι ιδανικός για τον έλεγχο ροής σε δίκτυα on-chip.

1.4 Εικονικά κανάλια (Virtual Channels)

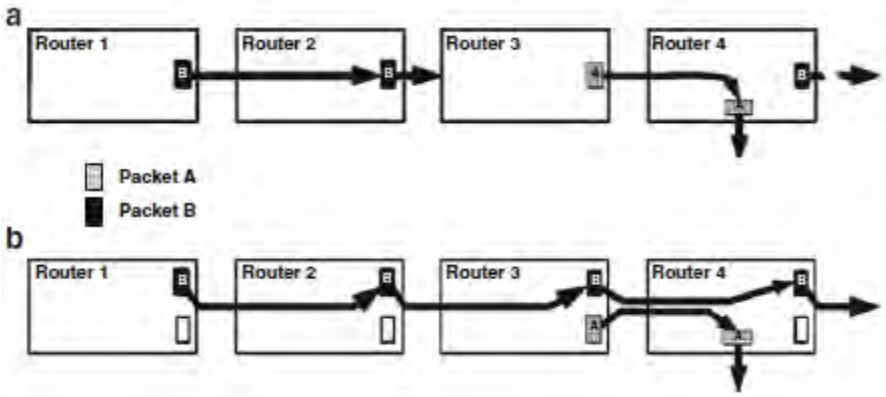
Στα δίκτυα wormhole υπάρχει πιθανότητα αποκλεισμού ενός πακέτου όταν αυτό προσπαθήσει να χρησιμοποιήσει ένα κανάλι το οποίο ταυτόχρονα χρησιμοποιείτε και από άλλο πακέτο είναι υπαρκτή. Με χρήση των εικονικών

καναλιών (VCs) μπορούμε να αποκλείσουμε την πιθανότητα αποκλεισμού ενός πακέτου. Αυτό επιτυγχάνετε με την εκχώρηση πολλών VCs, το καθένα σε ξεχωριστή ουρά flit, σε κάθε φυσικό κανάλι. Για κάθε VC, όταν φτάσει η κεφαλίδα του flit, το εισερχόμενο πακέτο εκχωρείτε στο buffer και παραμένει εκεί μέχρι να μεταδοθεί όλο το πακέτο.

Όπως φαίνεται στην εικόνα 1.3 το εισερχόμενο flit σπάει και αποθηκεύεται σε ξεχωριστά buffer τα οποία ενώνονται και πάλι στην έξοδο. Αν ένα κανάλι αποκλειστεί για κάποιο λόγο, τότε τα άλλα κανάλια είναι ικανά να μεταφέρουν το πακέτο στην έξοδο. Τα VCs δημιουργήθηκαν για να βελτιώσουν την απόδοση ενός δικτύου και να βελτιώσουν την καθυστέρηση στη μετάδοση των δεδομένων μας.



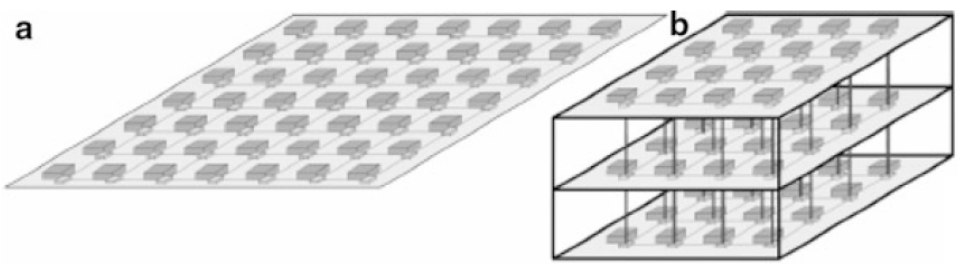
Εικόνα 1.3 Τυπικός δρομολογητής με VC



Εικόνα 1.4 Χρησιμοποίηση VC για αποφυγή deadlock

1.5 Διαστάσεις Δικτύου

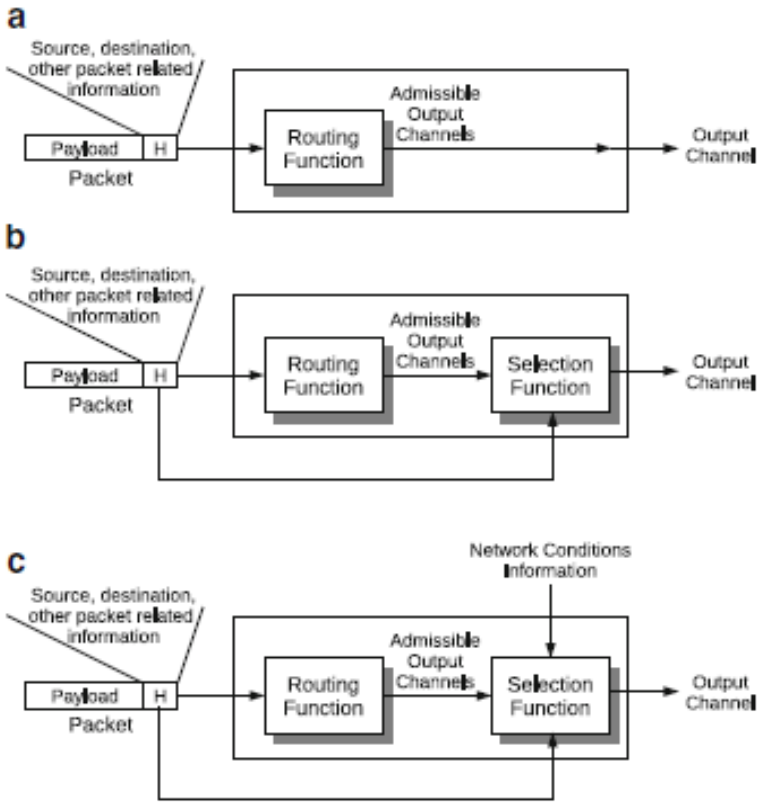
Στα NoC συναντάμε δύο βασικές αρχιτεκτονικές: α) Δύο διαστάσεων (2-D) και β) Τριών διαστάσεων (3-D). Όπως φαίνεται και στην εικόνα 1.5.α οι πυρήνες στα 2-D NoC βρίσκονται σε ένα στρώμα. Ενώ στα 3-D NoC (εικόνα 1.5.β) τα στρώματα στοιβάζονται το ένα πάνω στο άλλο [8], [9]. Σε κάθε στρώμα έχουμε τη δυνατότητα να χρησιμοποιήσουμε διαφορετική τεχνολογία, τοπολογία, συχνότητα ρολογιού κ.α. Τα τελευταία χρόνια μέσω της through-silicon-via (TSV) τεχνολογία έχουμε καταφέρει να συνδέσουμε αποτελεσματικά τα στρώματα των NoC μεταξύ τους (vertical channels). Η TSV κάνει την σύνδεση των στρωμάτων γρηγορότερη και αποτελεσματικότερη. Η εικόνα 1.5 απεικονίζει ένα 2-D και ένα 3-D NoC με παρόμοιο αριθμό πυρήνων.



Εικόνα 1.5 α) 2-D NoC β) 3-D NoC

1.6 Αλγόριθμοι Δρομολόγησης

Δρομολόγηση είναι η διαδικασία προώθησης των πακέτων μέσω των «κατάλληλων κόμβων από την πηγή στον τελικό προορισμό τους. Σε γενικές γραμμές ένας αλγόριθμος δρομολόγησης μπορεί να θεωρηθεί ως μια αλληλουχία δύο βασικών αξόνων που υλοποιούν την λειτουργία δρομολόγησης [10], [11], [12] και τη λειτουργία επιλογής [13], [14], [15], όπως αναπαριστάτε στην εικόνα 1.6.



Εικόνα 1.6 Διαδικασία δρομολόγησης και επιλογής

Πρώτα απ' όλα μέσα από τη διαδικασία δρομολόγησης υπολογίζουμε το σύνολο των διαθέσιμων καναλιών από τα οποία το πακέτο μπορεί να φτάσει στον προορισμό του. Στη συνέχεια μέσα από τη διαδικασία δρομολόγησης επιλέγουμε τη βέλτιστη δυνατή διαδρομή που πρέπει να ακολουθήσει το κάθε

πακέτο δεδομένου των παραμέτρων – συνθηκών. Ο αλγόριθμος δρομολόγησης δεν επηρεάζει στο σύστημά μας μόνο το χρόνο μετάδοσης των δεδομένων, αλλά επηρεάζει επίσης τόσο την κατανάλωση ενέργειας, όσο και τη συμφόρηση του δικτύου μας.

1.6.1 Βασικές τεχνικές δρομολόγησης

Ο αλγόριθμος δρομολόγησης μπορεί να εφαρμοστεί τόσο στο δρομολογητή που βρίσκετε στην πηγή, όσο και μέσω ενός κατανεμημένου συστήματος να διαμοιραστεί στους δρομολογητές κατά μήκος της διαδρομής. Κατά την διαδικασία δρομολόγησης πηγής, ο αρχικός δρομολογητής επιλέγει την διαδρομή που θα ακολουθήσει το πακέτο και την ενσωματώνει στην επικεφαλίδα του κάθε πακέτου. Καθώς το πακέτο μετακινείται κατά μήκος της διαδρομής, οι πληροφορίες που βρίσκονται στην επικεφαλίδα διαβάζονται από τους ενδιάμεσους δρομολογητές και κατευθύνουν ανάλογα το πακέτο προς τον προορισμό. Ένα τέτοιο είδους σύστημα είναι μια απλή λύση για τα on-chip δίκτυα, αλλά το βασικό μειονέκτημά του είναι η συσσώρευση πληροφορία δρομολόγησης που στοιβάζεται στην επικεφαλίδα του πακέτου μιας και σε ένα δίκτυο διαμέτρου k χρειάζονται τουλάχιστον k πληροφορίες δρομολόγησης αποθηκευμένες σε κάθε πακέτο. Έτσι αν το δίκτυο μεγαλώσει αρκετά το μέγεθος της επικεφαλίδας αυξάνεται αναλογικά με αποτέλεσμα να καθιστάτε μη πρακτική για το σύστημά μας. Σε αντίθεση με την κατανεμημένη προσέγγιση δρομολόγησης όπου ο κάθε δρομολογητής μεμονωμένα λαμβάνει την κατάλληλη δρομολόγηση του πακέτου (ανάλογα με τις παραμέτρους), ενώ η επικεφαλίδα περιλαμβάνει μόνο τη διεύθυνση προορισμού. Κάθε ενδιάμεσος δρομολογητής εξετάζει την διεύθυνση προορισμού (σε ορισμένες περιπτώσεις είναι απαραίτητη και η διεύθυνση πηγής) και αποφασίζει μέσω ποιου καναλιού θα προωθήσει το πακέτο. Παρόλα αυτά η πολυπλοκότητα του συστήματος σε κάθε ενδιάμεσο δρομολογητή αυξάνει σε σχέση με τον προηγούμενο.

1.6.2 Ντετερμινιστική και προσαρμοστική δρομολόγηση

Τα κατανεμημένα συστήματα δρομολόγησης μπορούν να χαρακτηριστούν ως deterministic, oblivious και adaptive. Τα πακέτα στους ντετερμινιστικούς αλγορίθμους δρομολόγησής ακολουθούν συγκεκριμένη διαδρομή από το δρομολογητή πηγής στο δρομολογητή προορισμού. Μπορεί οι υλοποιήσεις των ντετερμινιστικών αλγορίθμων δρομολόγησης είναι απλές, αλλά σε καμιά περίπτωση αυτό δεν μπορεί να εξισορροπηθεί και να κατανεμηθεί ομοιόμορφα ο φόρτος μεταξύ των συνδέσεων [16], [17]. Για την αντιμετώπιση τέτοιων προβλημάτων χρησιμοποιούμε τους προσαρμοστικούς αλγορίθμους δρομολόγησης. Έτσι καθίστανται διαθέσιμες πολλαπλές διαδρομές μεταξύ πηγής και προορισμού, οι οποίες δεν εξαρτώνται από την τρέχουσα κατάσταση του δικτύου. Με την εξισορρόπησή του φόρτου κατά μήκος μιας διαδρομής, οι προσαρμοστικοί αλγόριθμοι βελτιώνουν την απόδοση και παράλληλα είναι πιο ανεκτικοί σε περιπτώσεις προβλήματος σε μία σύνδεση ή και σφάλμα σε κάποιο δρομολογητή. Με αυτό τον τρόπο μειώνεται η πιθανότητα ένα πακέτο να περάσει από ένα κόμβο που δυσλειτουργεί ή είναι κορεσμένος. Από την άλλη οι ντετερμινιστικοί αλγόριθμοι αποτελούν καλύτερη επιλογή για μια ομοιόμορφη μεταφορά των πακέτων μέσα από το δίκτυο. Τα πακέτα φθάνουν στον προορισμό τους με τη σειρά κάτι που δεν συμβαίνει στους προσαρμοστικούς αλγορίθμους δρομολόγησης μια και τα πακέτα ακολουθούν διαφορετικές διαδρομές. Έτσι απαιτείτε μια διαδικασία αναδιάταξης των πακέτων. Οι απαιτήσεις αυτές αυξάνουν τόσο την πολυπλοκότητα υλοποίησης του αλγορίθμου όσο και την πιθανότητα σφάλματος.

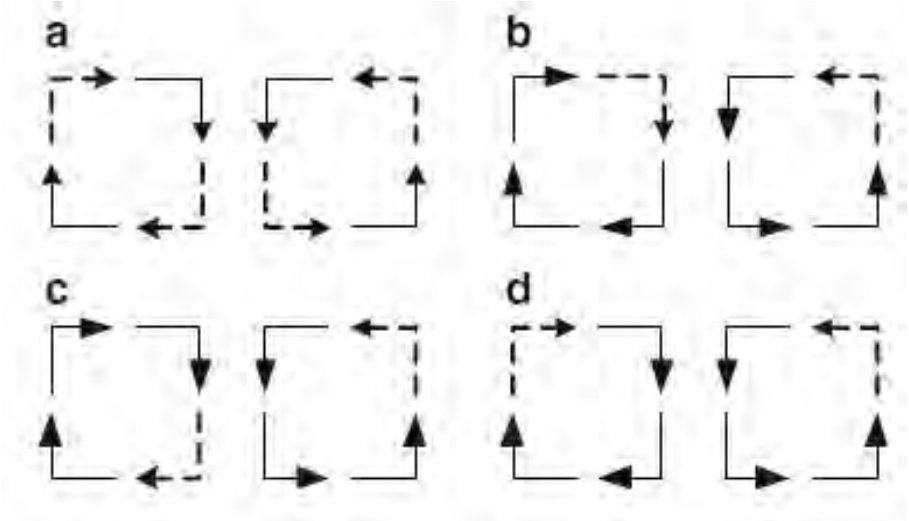
1.6.3 Unicast και Multicast πρωτόκολλά δρομολόγησης

Η επικοινωνία σε on-chip δίκτυα μπορεί να είναι είτε unicast (ένα προς ένα) είτε multicast/broadcast (ένα προς πολλά) [18], [19]. Στην δρομολόγηση unicast τα πακέτα στέλνονται σε ένα δρομολογητή προορισμού, ενώ στην

multicast επικοινωνία τα πακέτα στέλνονται σε ένα αυθαίρετο σύνολο δρομολογητών προορισμού. Έτσι η unicast επικοινωνία αποτελεί μια ειδική περίπτωση της multicast. Η multicast χρησιμοποιείτε συχνά σε πολλές εφαρμογές MPSoC όπως η replication [20], ο συγχρονισμός φράγματος [21], η συνοχή στης μνήμης cache σε κατανεμημένες αρχιτεκτονικές κρυφής μνήμης [22] και στο συγχρονισμό του ρολογιού [23]. Αν και η multicast επικοινωνία μπορεί να αναπαραχθεί από πολλές unicast μαζί κάτι τέτοιο δεν είναι πρακτικά εφικτό μιας και κάτι τέτοιο θα αύξανε σημαντικά την κυκλοφορία περιττής πληροφορίας στο σύστημά μας, αυξάνοντας σημαντικά την καθυστέρηση και την συμφόρηση στο δίκτυο [24].

1.6.4 Turn Model Routing

Το σύστημα εναλλαγής δρομολόγησης στα συστήματα NoC βασίζονται στο μηχανισμό εναλλαγής wormhole παρέχοντας πλήρη ελευθερία κινήσεων σε δισδιάστατα πλέγματα τοπολογίας mesh [11]. Αυτό το μοντέλο έχει επιλεγεί ως «εκπρόσωπος» της ελάχιστης και της μερικής προσαρμοστικής δρομολόγησης. Στο μοντέλο αυτό το deadlock [25] μπορεί να αποφευχθεί με την απαγόρευση κάποιων διαδρομών. Οι τέσσερεις πιο γνωστοί τύποι του συγκεκριμένου μοντέλου είναι ο XY, ο Negative-First (NF), ο West-First (WF) και ο North-Last (NL) όπως απεικονίζονται και στην εικόνα 1.7.



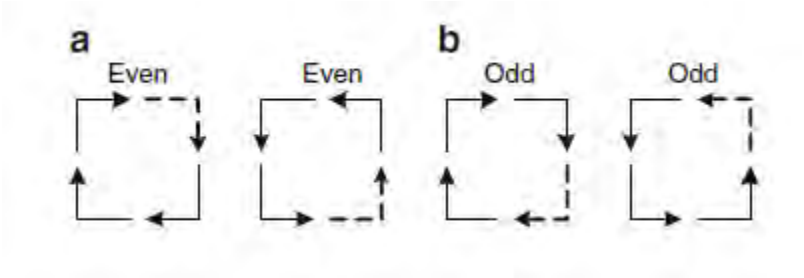
Εικόνα 1.7 Όλες οι επιτρεπτές εναλλαγές του turn model στον α) αλγόριθμο δρομολόγησης XY, β) στον Negative-First, γ) στον West-First και δ) στον North-Last (οι συνεχόμενές γραμμές δείχνουν τις επιτρεπόμενες διαδρομές (turns) ενώ με διακεκομμένες οι μη επιτρεπόμενες)

Παρά το γεγονός ότι ο αλγόριθμος δρομολόγησης XY απαγορεύει τέσσερεις στροφές για να αποφευχθεί το deadlock, τα άλλα μοντέλα απαγορεύουν μόνο δύο από τις οκτώ.

Το μοντέλο Odd-Even είναι επίσης ένα από τα αποδοτικότερα μοντέλα στους partial adaptive wormhole αλγόριθμους δρομολόγησης των δισδιάστατων mesh on-chip διασυνδέσεων [26] χωρίς εικονικά κανάλια. Σε αντίθεση με το turn model, το μοντέλο Odd-Even απαγορεύει συγκεκριμένες στροφές στις άρτιες και διαφορετικές στις περιττές στήλες. Ως εκ τούτου ο βαθμός προσαρμοστικότητας του συγκεκριμένου μοντέλου είναι υψηλότερος από τα υπόλοιπα μοντέλα. Το μοντέλο Odd-Even περιγράφετε από τους παρακάτω κανόνες:

Κανόνας 1: Τα East Turns δεν μπορούν να πραγματοποιηθούν στις άρτιες στήλες (εικόνα 1.8.α)

Κανόνας 2: Τα North Turns δεν μπορούν να πραγματοποιηθούν στις περιττές στήλες (εικόνα 1.8.β)

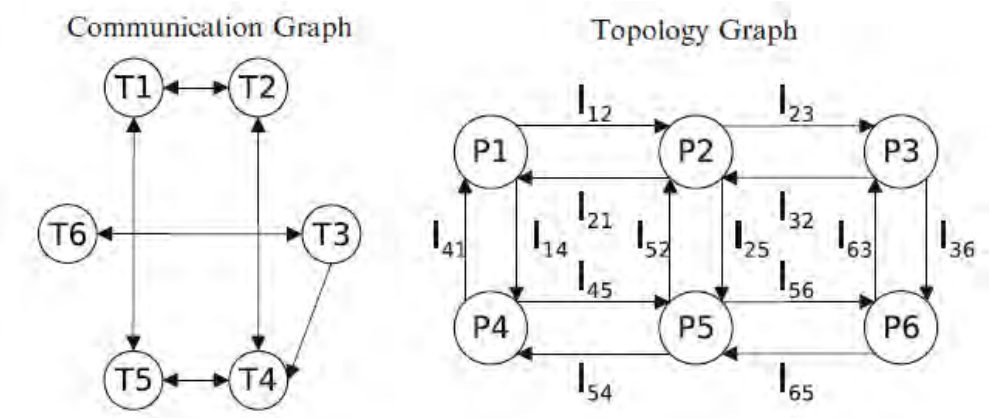


Εικόνα 1.8 Κανόνες μοντέλου Odd-Even α) απαγορευμένες διαδρομές στις άρτιες στήλες, β) απαγορευμένες διαδρομές στις περιττές στήλες

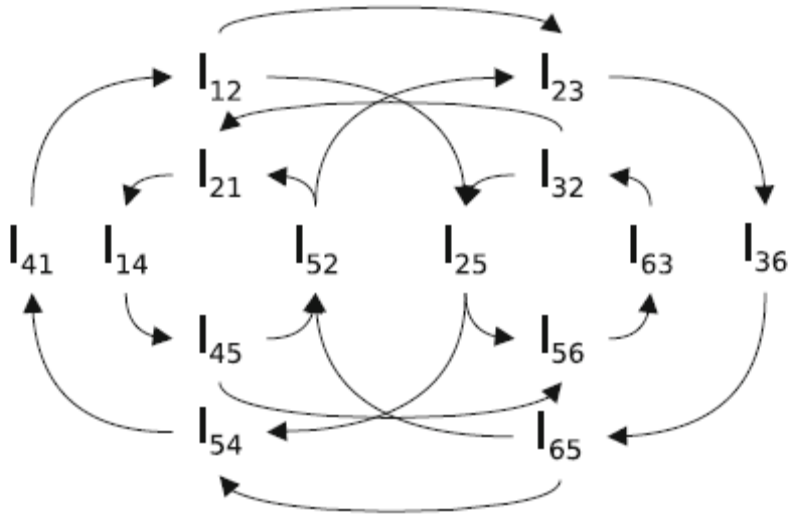
1.7 Εφαρμογή ειδικών αλγορίθμων δρομολόγησης

Σε αντίθεση με τα κλασσικά δίκτυα, στα on-chip δίκτυα η υπολογιστική ισχύς καθώς και οι απαιτήσεις της επικοινωνίας των εφαρμογών μπορούν να κατηγοριοποιηθούν ευκολότερα. Συγκριμένα στα συστήματα NoC είναι προαποφασισμένο ποια ζεύγη πυρήνων θα επικοινωνήσουν μεταξύ τους και ποια όχι. Θεωρητικά μπορεί κάποιος να εκτιμήσει αλλά και να αναλύσει ορισμένα ποσοτικά στοιχεία, όπως οι απαιτήσεις του εύρους ζώνης μεταξύ των πυρήνων. Αφού χαρτογραφηθεί και προγραμματιστεί ένα NoC , έχουμε όλες τις απαραίτητες πληροφορίες για να προχωρήσουμε σε μια περαιτέρω ανάλυση και προσομοίωση του συστήματός μας. Η μεθοδολογία APplication Specific Routing Algorithms (APSRA) [27] επιτρέπει μέσω μιας ειδικής εφαρμογής τη δημιουργία αλγορίθμων υψηλής απόδοσης. Η βασική ιδέα πίσω από το APSRA είναι μέσω ενός channel dependency graph (CDG) [28], να εξετάσουμε μόνο τις άμεσες εξαρτήσεις των καναλιών που δημιουργούνται μεταξύ των πύρινων. Έτσι μέσω της εφαρμογής Application Specific CDG (ASCDG), βρίσκουμε το CDG που εκτελείται σε λιγότερους κύκλους, βελτιώνοντας τη λειτουργία της δρομολόγησης.

Ας θεωρήσουμε το γράφημα της εικόνας 1.9. Για λόγους απλότητας θεωρούμε ότι η διεργασία Ti αντιστοιχίζεται σε κόμβο Pi, i = 1, 2, …., 6



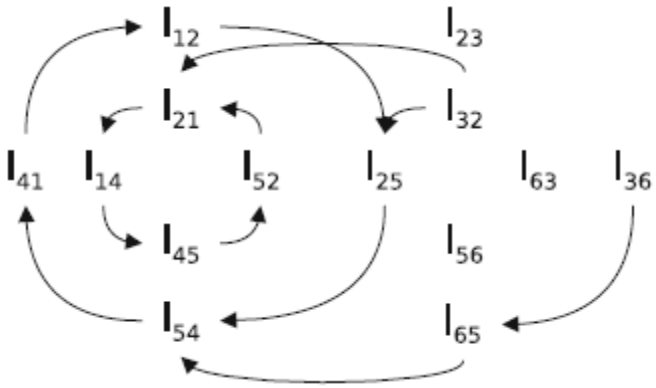
Εικόνα 1.9 Γράφοι Τοπολογίας και επικοινωνίας



Εικόνα 1.10 Γράφημα εξαρτήσεως καναλιού της τοπολογίας δικτύου της εικόνας 1.9, για οποίο δίνετε ένας ελάχιστος και πλήρης προσαρμοστικός αλγόριθμος δρομολόγησης

Ο CDG είναι ένας πλήρως προσαρμοστικός αλγόριθμος δρομολόγησης και απεικονίζεται στην εικόνα 1.10. Δεδομένου ότι περιέχει πολλούς κύκλους, σύμφωνα με το θεώρημα του Duato [28] δεν μπορεί να επιβεβαιώσει το deadlock free για την συγκεκριμένη τοπολογία. Για να δημιουργήσουμε το deadlock free θα πρέπει να σπάσουμε όλους τους κύκλους του CDG. Το να σπάσουμε όμως έναν κύκλο μέσω της αφαίρεσης μιας εξάρτησης έχει ως αποτέλεσμα τον περιορισμό της λειτουργίας της δρομολόγησης και κατά συνέπεια την απώλεια της προσαρμοστικότητάς της. Καθώς αφαιρούμε ολοένα και περισσότερους κύκλους, η προσαρμοστικότητα του αλγορίθμου μας όσον αφορά το deadlock θα μειωθεί σε μεγάλο βαθμό.

Είναι προφανές ότι μπορούμε να αφαιρέσουμε πολλούς κύκλους του CDG χωρίς να επηρεάσουμε την προσαρμοστικότητα του αλγορίθμου δρομολόγησης. Ας εξετάσουμε για παράδειγμα την εξάρτηση $i_{1,2} \rightarrow i_{2,3}$. Παρατηρούμε ότι εμφανίζεται στο CDG αλλά όχι και στον ASCDG. Στην πραγματικότητα τα κανάλια $i_{1,2}$ και $i_{2,3}$ μπορούν να χρησιμοποιηθούν μόνο στις ακολουθίες $T_1 \rightarrow T_3$, $T_1 \rightarrow T_6$ και $T_4 \rightarrow T_3$ οι οποίες δεν υπάρχουν στο γράφημα της επικοινωνίας. Αν αναλύσουμε τις υπόλοιπες εξαρτήσεις, λαμβάνοντας υπόψη το γράφημα της επικοινωνίας, διαπιστώνουμε ότι μπορούμε να αφαιρέσουμε πολλές από αυτές τις εξαρτήσεις χωρίς την ανάγκη περιορισμού της λειτουργίας του αλγορίθμου δρομολόγησης. Το προκύπτον ASCDG απεικονίζεται στην εικόνα 1.11. Έχει προκύψει από το CDG αφαιρώντας όλες τις εξαρτήσεις που δεν μπορούν να ενεργοποιηθούν από το γράφημα της επικοινωνίας.



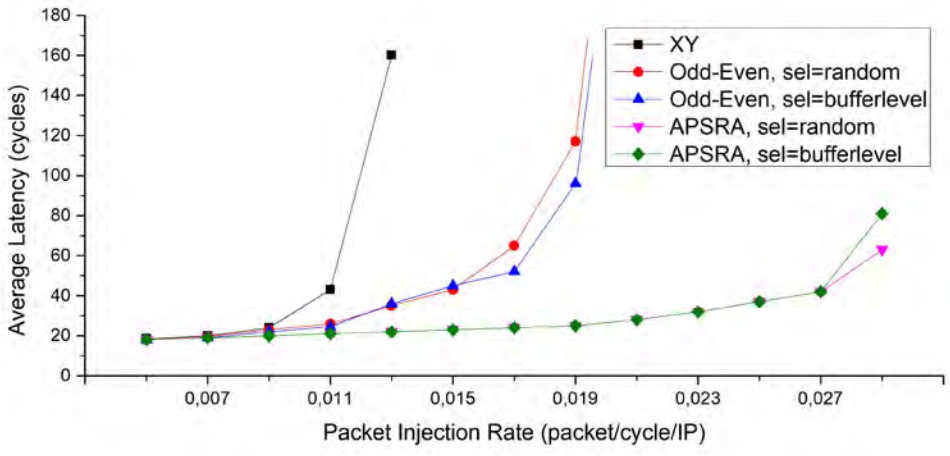
Εικόνα 1.11 Εφαρμογή συγκεκριμένου καναλιού στο γράφημα εξάρτησης για την τοπολογία δικτύου της εικόνας 1.10

Αν και ο αριθμός των κύκλων μειώνεται σε δύο στην ASCDG, παραμένει το ενδεχόμενο ενός deadlock. Για να αποφύγουμε ένα τέτοιο ενδεχόμενο μπορούμε να σπάσουμε τους κύκλους ως εξής. Η εξάρτηση του καναλιού $i_{4,1} \rightarrow i_{1,2}$ οφείλεται στην επικοινωνία $T_4 \rightarrow T_2$. Αυτή η επικοινωνία μπορεί να πραγματοποιηθεί και εναλλακτικά από διαφορετικά μονοπάτια όπως $P_4 \rightarrow P_5 \rightarrow P_2$ και $P_4 \rightarrow P_1 \rightarrow P_2$. Αν και η λειτουργία δρομολόγησης περιορίζεται κατ' αυτόν τον τρόπο, η εφαρμογή της εξάρτησης του καναλιού από $i_{4,1} \rightarrow i_{1,2}$ δεν υφίσταται πλέον. Με παρόμοιο τρόπο μπορούμε να σπάσουμε και τον δεύτερο κύκλο, για παράδειγμα, με την αφαίρεση της εξάρτησης $i_{1,4} \rightarrow i_{4,5}$ λόγο της επικοινωνίας $T_1 \rightarrow T_5$.

1.8 Απόδοση συστήματος

Πολλών ειδών μετρήσεις έχουν χρησιμοποιηθεί για την εκτίμηση, την αξιολόγηση και την σύγκριση των επιδόσεων ενός NoC. Οι πιο συχνές μετρήσεις που χρησιμοποιούνται είναι η μέση καθυστέρηση και η μέση απόδοση. Η μέση καθυστέρηση είναι η μέση τιμή του μέσου όρου της

καθυστέρησης επικοινωνίας σε όλες τις επικοινωνίες. Η μέση καθυστέρηση είναι ο μέσος χρόνος που χρειάζεται ένα πακέτο για να φθάσει στον προορισμό του. Η καθυστέρηση των πακέτων είναι ο χρόνος που παρέρχεται μεταξύ την εισχώρησης της επικεφαλίδας του πακέτου στο δίκτυο και τον επιτυχή παραλαβή της ουράς του πακέτου από τον κόμβο προορισμού. Η μέση απόδοση είναι ο μέσος όρος της απόδοσης των κόμβων προορισμού. Δηλαδή ο μέσος όρων των πακέτων που λαμβάνονται από τον προορισμό στην μονάδα του χρόνου. Συνήθως η μέση καθυστέρηση και η μέση απόδοση αποδίδονται μέσω διαγραμμάτων για διαφορετικούς ρυθμούς εισαγωγής των πακέτων. Παράδειγμα ενός τέτοιου διαγράμματος φαίνεται στην εικόνα 1.12.



Εικόνα 12 Ενδεικτική χρονοκαθυστέρηση αλγορίθμων δρομολόγησης

1.9 Συμπεράσματα

Σε αυτό το εισαγωγικό κεφάλαιο παρουσιάστηκαν βασικές έννοιες σχεδιασμού ενός NoC για διάφορες τοπολογίες άμεσης και έμμεσης δρομολόγησης. Διαφορετικές διατάξεις μεταγωγής πακέτων, ροής μηχανισμών ελέγχου, χρήση εικονικών καναλιών, σχεδιαγραμμάτων δρομολόγησης, τεχνικών επιλογής εξόδου και μια γενική περιγραφή της

αρχιτεκτονικής ενός Network-on-Chip. Στα κεφάλαια που ακολουθούν γίνετε μια λεπτομερής περιγραφή κατηγοριών συστημάτων NoC που παρουσιάζουν εξαιρετικό ενδιαφέρον.

Βιβλιογραφία

1. L. Benini, G.D.Micheli, Networks on chips: a new SoC paradigm. IEEE Comput. **35**(1), 70–78 (2002)

2. W.J. Dally, B. Towles, Route packets, not wires: on-chip interconnection networks, in *ACM/IEEE Design Automation Conference*, Las Vegas, 2001, pp. 684–689

3. S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, A. Hemani, A network on chip architecture and design methodology, in *IEEE Computer Society Annual Symposium on VLSI*, Pittsburg, p. 117, 2002

4. G.D. Micheli, L. Benini, Powering networks on chips: energy-efficient and reliable interconnect design for SoCs, in *International IEEE Symposium on Systems Synthesis*, Montréal, 2001, pp. 33–38

5. J. Duato, S. Yalamanchili, L. Ni, *Interconnection Networks: An Engineering Approach* (Morgan Kaufmann, San Francisco, 2002)

6. A. Jantsch, H. Tenhunen (eds.), *Networks on Chip*, chapter 1 (Kluwer Academic, Boston, 2003)

7. P. Mohapatra, Wormhole routing techniques for directly connected multicomputer systems. ACM Comput. Surv. **30**(8), 374–410 (1998)

8. C.-H. Chao, K.-Y. Jheng, H.-Y.Wang, J.-C.Wu, A.-Y.Wu, Traffic- and thermal-aware run-time thermal management scheme for 3D NoC systems, in

ACM/IEEE International Symposium on Networks-on-Chip, Grenoble, 2010, pp. 223–230

9. M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, H. Tenhunen, Cluster-based topologies for 3D networks-on-chip using advanced inter-layer bus architecture. Elsevier J. Comput. Syst. Sci. **79**(4), 475–491 (2013)

10. A.A. Chien, J.H. Kim, Planar-adaptive routing: low-cost adaptive networks for multiprocessors. J. ACM **42**(1), 91–123 (1995)

11. C.J. Glass, L.M. Ni, The turn model for adaptive routing. J. Assoc. Comput. Mach. **41**(5), 874–902 (1994)

12. J. Upadhyay, V. Varavithya, P. Mohapatra, A traffic-balanced adaptive wormhole routing scheme for two-dimensional meshes. IEEE Trans. Comput. **46**(2), 190–197 (1997)

13. G. Ascia, V. Catania, M. Palesi, D. Patti, Implementation and analysis of a new selection strategy for adaptive routing in networks-on-chip. IEEE Trans. Comput. **57**(6), 809–820 (2008)

14. J. Hu, R. Marculescu, DyAD – smart routing for networks-on-chip, in *ACM/IEEE Design Automation Conference*, San Diego, 7–11 June 2004, pp. 260–263

15. E. Nilsson, M. Millberg, J. Oberg, A. Jantsch, Load distribution with the proximity congestion awareness in a network on chip, in *Design, Automation and Test in Europe*, Washington, D.C., 2003, pp. 1126–1127

16. D.P. Bertsekas, R.G. Gallager, *Data Networks* (Prentice Hall, Englewood Cliffs, 1992)

17. W.J. Dally, B. Towles, *Principles and Practices of Interconnection Networks* (Morgan Kaufmann, San Francisco, 2004)

18. M. Daneshtalab, M. Ebrahimi, T.C. Xu, P. Liljeberg, H. Tenhunen, A generic adaptive pathbased routing method for MPSoCs. Elsevier J. Syst. Archit. **57**(1), 109–120 (2011)

19. M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, J. Flich, H. Tenhunen, Path-based partitioning methods for 3d networks-on-chip with minimal adaptive routing. IEEE Trans. Comput. **99**, pp. 1, doi: 10.1109/TC.2012.255

20. P.K. McKinley, H. Xu, E.T. Kalns, L.M. Ni, CompaSS: efficient communication services for scalable architectures, in *International Conference on Supercomputing*, Washington, D.C., 1992, pp. 478–487

21. H. Xu, P.K. McKinley, E.T. Kalns, L.M. Ni, Efficient implementation of barrier synchronization in wormhole-routed hypercube multicomputers. J. Parallel Distrib. Comput. **16**, 172–184 (1992)

22. K. Li, R. Schaefer, A hypercube shared virtual memory, in *International Conference on Parallel Processing*, University Park, 1989, pp. 125–132

23. M.M. de Azevedo, D. Blough, Fault-tolerant clock synchronization of large multicomputers via multistep interactive convergence, in *International Conference on Distributed Computing Systems*, Hong Kong, 1996, pp. 249–257

24. X. Lin, L.M. Ni, Multicast communication in multicomputer networks. IEEE Trans. Parallel Distrib. Syst. **4**, 1105–1117 (1993)

25. L.M. Ni, P.K. McKinley, A survey of wormhole routing techniques in direct networks. IEEE Comput. **26**, 62–76 (1993)

26. G.-M. Chiu, The odd-even turn model for adaptive routing. IEEE Trans. Parallel Distrib. Syst. **11**(7), 729–738 (2000)

27. M. Palesi, R. Holsmark, S. Kumar, V. Catania, Application specific routing algorithms for networks on chip. IEEE Trans. Parallel Distrib. Syst. **20**(3), 316–330 (2009)

28. J. Duato, A necessary and sufficient condition for deadlock-free routing in wormhole networks. IEEE Trans. Parallel Distrib. Syst. **6**(10), 1055–1067 (1995)

$2^{ο}$ ΚΕΦΑΛΑΙΟ

ADAPTIVE ROUTING PROTOCOLS IN NOC

2.1 Εισαγωγή

Τα δίκτυα on-chip, όπως και τα δίκτυα υπολογιστών, μπορούν να επωφεληθούν από την "συσκευασία" των δεδομένων σε πακέτα [1][2]. Δεδομένου ότι τα δίκτυα on-chip χρησιμοποιούν ελαφρύτερα και πιο γρήγορα πρωτόκολλα δεν είναι απαραίτητο να ακολουθήσουν όλα τα πρότυπα των δικτύων που χρησιμοποιούνται στους υπολογιστές.

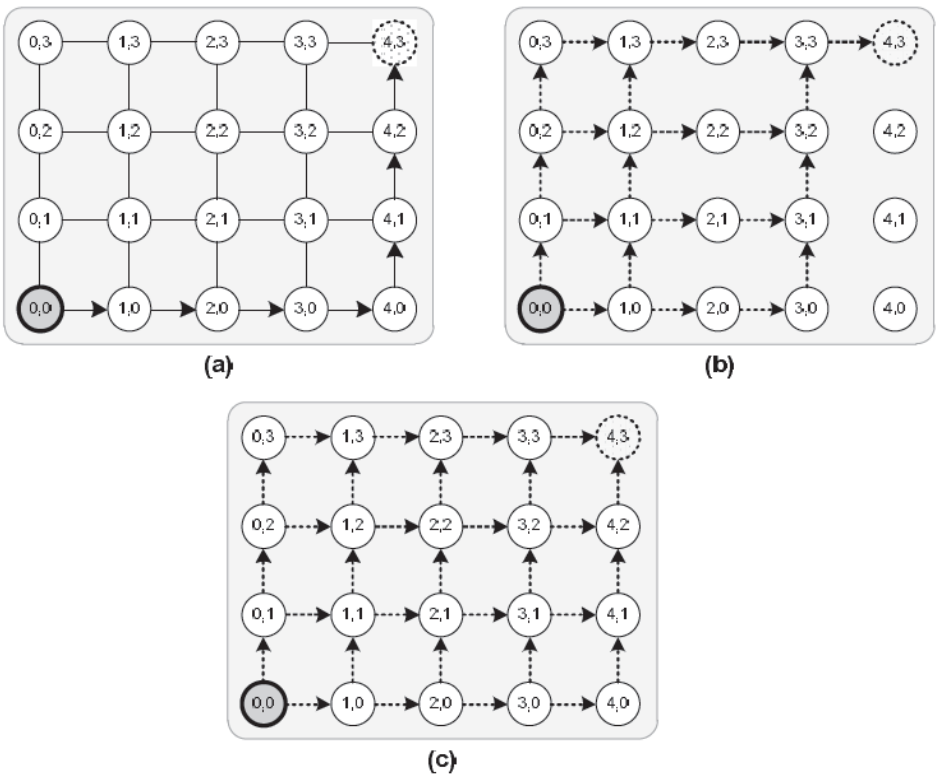
Η τοπολογία mesh είναι η πλέον κατάλληλη για τη διαχείριση και προώθηση πακέτων σε συστήματα NoC που χρησιμοποιούν adaptive αλγόριθμους δρομολόγησης. Μια τοπολογία mesh 2D $m \times n$ αποτελείται από $N(= m \times n)$ κόμβους, όπου ο κάθε κόμβος χαρακτηρίζεται από ένα ζεύγος ακεραίων συντεταγμένων (x, y) τέτοιο ώστε $0 \le x \le n$ και $0 \le y \le m$. Δύο κόμβοι με συντεταγμένες (x_i, y_i) και (x_j, y_j) συνδέονται μέσω ενός καναλιού επικοινωνίας μόνο εάν $| x_i - x_j | + | y_i - y_j | = 1$.

2.2 Unicast routing protocols

Στα Noc, οι αλγόριθμοι δρομολόγησης χρησιμοποιούνται για τον προσδιορισμό μιας διαδρομής από τον κόμβο πηγής στον κόμβο προορισμού. Κάποιοι από τους σημαντικότερους unicast αλγορίθμους δρομολόγησης παρουσιάζονται παρακάτω.

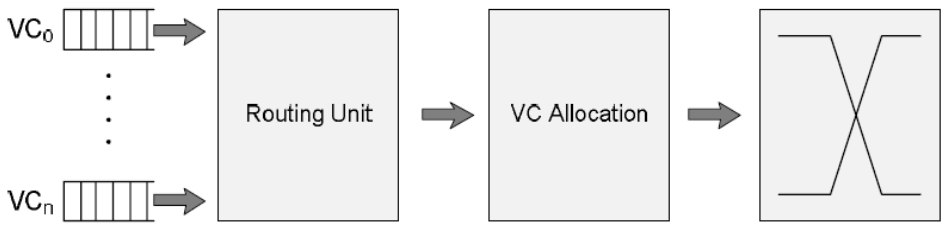
2.2.1 XY δρομολόγηση

Ο XY, είναι ένας αλγόριθμος δρομολόγησης που χρησιμοποιείται κυρίως σε αρχιτεκτονικές mesh [3][4][5]. Όπως φαίνεται στην εικόνα 2.1.α σε αυτόν τον αλγόριθμο κάθε πακέτο μετακινείται πρώτα κατά μήκος του X και μετά κατά τη διεύθυνση Y για να φτάσει στον προορισμό του.



Εικόνα 2.1 Σχηματική αναπαράσταση τριών διαφορετικών περιπτώσεων δρομολόγησης adaptive αλγορίθμων

Εκτός από τους ακριανούς δρομολογητές, οι υπόλοιποι XY δρομολογητές έχουν 5 θύρες εισόδου/εξόδου (τέσσερεις συνδέονται με τους γειτονικούς δρομολογητές και ένας με τον πυρήνα). Τα κύρια αρχιτεκτονικά στοιχεία ενός XY δρομολογητή περιλαμβάνουν την είσοδο FIFO για κάθε θύρα, τη μονάδα υπολογισμού της διαδρομής, την μονάδα κατανομής εικονικού καναλιού (VC) (εάν υπάρχει) και τη λογική μονάδα crossbar. Για να ελαχιστοποιηθεί η καθυστέρηση και οι απαιτούμενοι πόροι, χρησιμοποιείται η μέθοδος wormhole για τη μεταγωγή. Ένα flit εισέρχεται στο δρομολογητή μέσω μια από τις θύρες στο FIFO. Αν το flit είναι μια κεφαλίδα, υποδεικνύοντας την ύπαρξη ενός νέου πακέτου, προχωράς στη μονάδα δρομολόγησης, η οποία καθορίζει τη θύρα εξόδου που πρέπει να χρησιμοποιήσει το πακέτο. Το flit header προσπαθεί να αποκτήσει ένα κανάλι για το επόμενο hop. Μέσω του crossbar το flit διασχίζει το δρομολογητή και εισέρχεται στο κανάλι. Μεταγενέστερα flits που ανήκουν στο ίδιο πακέτο μπορούν να προχωρήσουν κατευθείαν στο crossbar και στη συνέχεια στη θύρα εξόδου. Το κύριο αρχιτεκτονικό στοιχείο ενός δρομολογητή XY απεικονίζεται στην εικόνα 2.2.

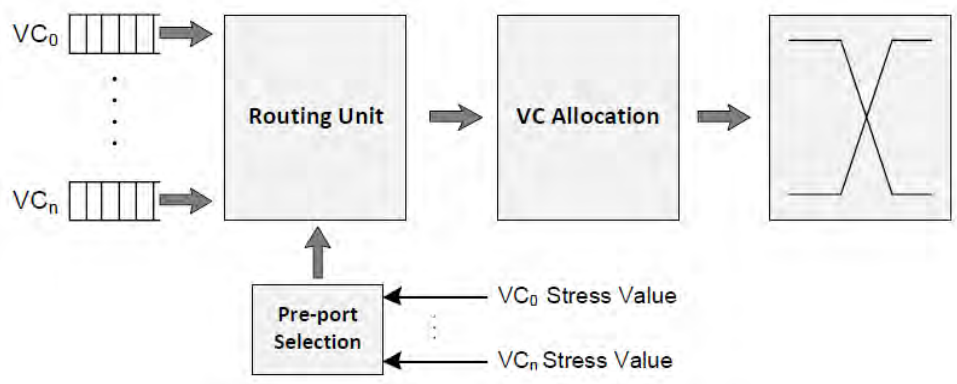


Εικόνα 2.2 Δομή ενός δρομολογητή XY

2.2.2 Δρομολόγηση DyAD

Το μοντέλο Odd-Even παρουσιάστηκε στο 1^{ο} κεφάλαιο, όπου τα πακέτα στρέφονται προς τους περριτούς αριθμούς δρομολογητών. Αυτό καθιστά την τεχνική ως μερικώς adptive σύστημα χωρίς αδιέξοδο που χρησιμοποιεί ελάχιστες διαδρομές.

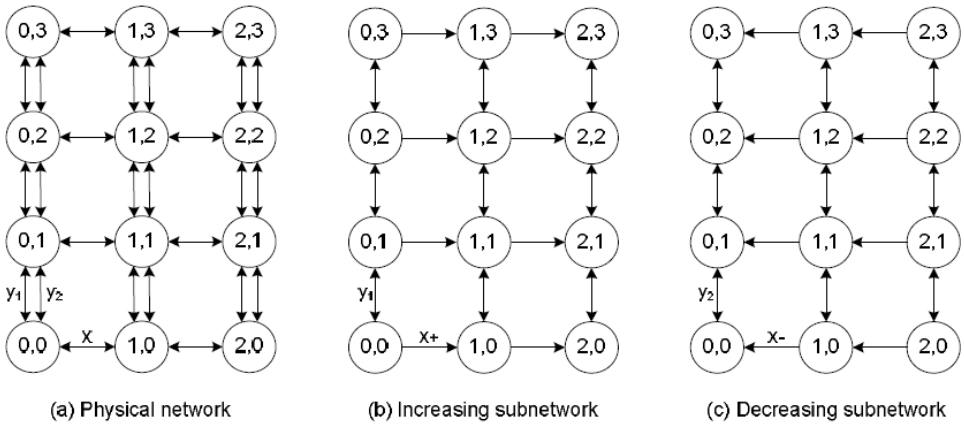
Δεδομένου ότι ο συγκεκριμένος αλγόριθμος είναι deadlock-free, δεν υπάρχει η ανάγκη δημιουργία εικονικών καναλιών (για να αποφευχθεί το αδιέξοδο), αλλά μπορούν να δημιουργηθούν εικονικά κανάλια για να κερδίσει σε επιδώσεις. Λαμβάνοντας υπόψη το παράδειγμα της εικόνα 2.1.β παρουσιάζονται όλες οι πιθανές ελάχιστες διαδρομές (10 διαδρομές) για τα πακέτα από τον κόμβο πηγής (0, 0) στον κόμβο προορισμού (4, 3). Το DyAD (Dynamic Adaptive Deterministic Switching) είναι ένα μερικώς προσαρμοστικό σύστημα με βάση το Odd-Even. Είναι ένας συνδυασμός ντετερμινιστικών (XY) και προσαρμοστικών (Odd-Even) σχημάτων δρομολόγησης [6]. Πιο συγκεκριμένα όταν το δίκτυο δεν έχει συμφόρηση, ο δρομολογητής DyAD λειτουργεί ντετερμινιστικά και όταν στο δίκτυο έχει συμφόρηση ο δρομολογητής DyAD χρησιμοποιεί την προσαρμοστική λειτουργία δρομολόγησης και έτσι αποφεύγει υπερφορτωμένους συνδέσμους αξιοποιώντας άλλες διαδρομές δρομολόγησης. Ως εκ τούτου η κύρια διαφορά μεταξύ των δρομολογητών DyAD/Odd-Even και XY είναι ότι, ανάλογα με την κατάσταση δικτύου, η μονάδα δρομολόγησης μπορεί να επιλέξει διαφορετικές διαδρομές σε διαφορετικές χρονικές στιγμές για το ίδιο ζεύγος πηγής και προορισμού. Για να συμβεί αυτό προστίθεται στο δρομολογητή μια μονάδα επιλογής για να επιλέξει την βέλτιστη γειτονική θύρα και να στείλει τις πληροφορίες στην μονάδα δρομολόγησης (Εικόνα 3.3). Ένας από τους παράγοντες επιλογής μιας θύρας εξόδου είναι ο αριθμός του ελεύθερου buffer στην αντίστοιχη θύρα εξόδου στο επόμενο hop [7]. Αυτή η τεχνική έχει χρησιμοποιηθεί στα διάφορα συστήματα προσαρμοστικής δρομολόγησης, όπου χρησιμοποιείται για την μέτρηση του διαθέσιμου buffer στους κόμβους για τον υπολογισμό της συμφόρησης. Για να μεταφέρουμε τις πληροφορίες μετρήσεις προστίθενται κάποια καλώδια μεταξύ γειτονικών δρομολογητών.



Εικόνα 2.3 Δομή δρομολογητή DyAD

2.2.3 Δομή αλγορίθμου DyXY

Όπως ήδη έχουμε αναφέρει ο αλγόριθμος για τη μεταφορά πακέτων κατά μήκος της συντομότερης διαδρομής ονομάζεται πλήρως προσαρμοστικός. Ο αλγόριθμος δρομολόγησης Dynamic XY (DyXY) ως εκ τούτου χαρακτηρίζεται ως πλήρως προσαρμοστικός. Λόγω του γεγονότος ότι σε κάθε κόμβο τα πακέτα δρομολόγησης μπορούν να δρομολογηθούν και προς τις δύο κατευθύνσεις X και Y, αυτός ο αλγόριθμος απαιτεί την ύπαρξη μηχανισμού για την αποφυγή αδιεξόδου. Έτσι λοιπόν δημιουργούνται εικονικά κανάλια στην διάσταση Y τα οποία διαιρούνται σε δύο κομμάτια όπως φαίνεται στην εικόνα 2.4. Το δίκτυο είναι χωρισμένο σε δύο υποδίκτυα που ονομάζονται υποδίκτυα +X και –X, το καθένα από τα οποία έχει το ήμισυ των καναλιών στη διάσταση Y.



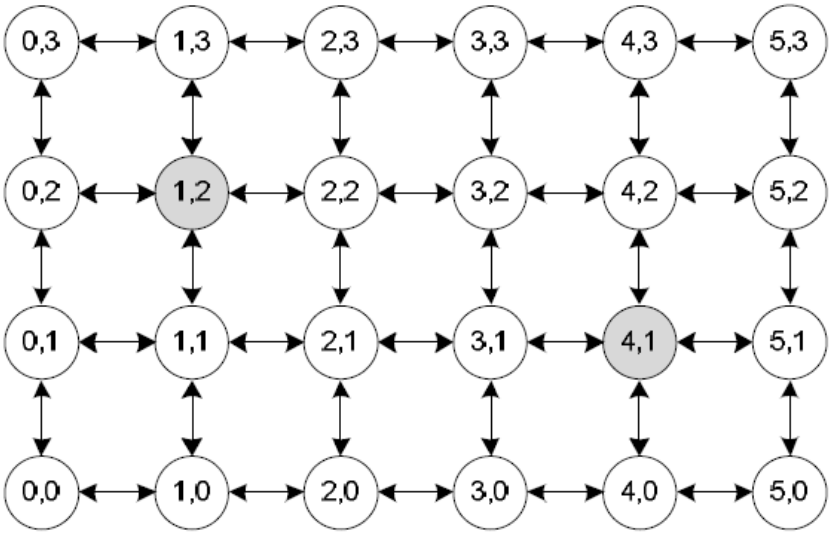
Εικόνα 2.4 α) 3X4 mesh δίκτυο και αντίστοιχη β) αύξηση γ) μείωση σε υποδίκτυα

Αν ο κόμβος προορισμού βρίσκεται δεξιά της πηγής το πακέτο θα δρομολογηθεί μέσω του υποδικτύου +X, ενώ αν ο κόμβος προορισμού βρίσκεται αριστερά της πηγής το πακέτο θα δρομολογηθεί μέσω του υποδικτύου –X. Σε διαφορετική περίπτωση το πακέτο θα δρομολογηθεί με ένα από τα δύο υποδίκτυα, για να αποφευχθεί το ενδεχόμενο ενός αδιεξόδου. Εφόσον τα υποδίκτυα είναι ακυκλικά, τα πακέτα μπορούν να δρομολογηθούν προσαρμοστικά κατά μήκος των συντομότερων διαδρομών που σημαίνει ότι για κάθε κόμβο ένα πακέτο μπορεί να δρομολογηθεί κατά μήκος κάθε διάστασης. Η εικόνα 2.1 απεικονίζει τις πιθανές ελάχιστες διαδρομές του DyXY. Δηλαδή έχουμε $\frac{(3+4)!}{3!\times 4!} = 35$ δυνατές διαδρομές [8]. Η δομή του δρομολογητή του DyXY φαίνεται στην εικόνα 2.3 και είναι πανομοιότυπος με αυτόν του DyAD.

Η βασική αδυναμία του DyXY είναι σε περιπτώσεις δρομολόγησης όπου ο κόμβος πηγής και προορισμού διαφέρουν μόνο κατά μια θέση είτε τη θέση X είτε την Y. Σε τέτοιες περιπτώσεις υπάρχει μεγάλη πιθανότητα ο αλγόριθμος μπορεί να οδηγήσει σε αδιέξοδο και κάποιος κόμβος να βρεθεί χωρίς να έχει επιλογή για το που θα στείλει το πακέτο.

2.2.4 Δομή αλγορίθμου EDXY

Στόχος του αλγορίθμου Enhanced Dynamic XY (EDXY) είναι να επιλύσει και να ενισχύσει τον αλγόριθμο DyXY. Αυτό επιτυγχάνεται χρησιμοποιώντας μια σημαία που υποδεικνύει συμφόρηση κατά μήκος μιας σειράς ή μιας στήλης. Αυτή η σημαία διαδίδεται κατά μήκος μιας σειράς (ή μιας στήλης) και υποδεικνύει στις παρακείμενες σειρές (ή στήλες) ότι αυτή η σειρά (ή στήλη) είναι σχεδόν κορεσμένη και πρέπει να αποφεύγεται. Καθώς η σημαία μεταδίδεται κατά μήκος μιας γραμμής (ή μιας στήλης) είναι προφανές ότι κάθε δρομολογητής μεταδίδει τη σημαία του προηγούμενου δρομολογητή συμφόρησης. Εάν το buffer ενός δρομολογητή ξεπεράσει την τιμή κατωφλίου, ο δρομολογητής θα ενεργοποιήσει τη σημαία συμφόρησής του. Προκειμένου να παρακολουθείται αποτελεσματικά η κατάσταση συμφόρησης προστίθενται δύο σημαίες σε κάθε σειρά (ή την στήλη) η μία ενημερώνει τους αριστερούς δρομολογητές συμφόρησης σε έναν δρομολογητή δεξιάς πλευράς και ο άλλος ενημερώνει τους δρομολογητές δεξιάς πλευράς για τη συμφόρηση σε έναν αριστερό δρομολογητή. Για το σκοπό αυτό μεταξύ δύο γειτονικών δρομολογητών προστίθενται δύο καλώδια συμφόρησης ένα σε κάθε κατεύθυνση. Τα καλώδια συμφόρησης ομαδοποιούνται με καλώδια τα οποία χρησιμοποιούνται για τη μετάδοση του ελεύθερου buffer (τιμή τάσης) στους επικαλυπτόμενους δρομολογητές. Όταν η σημαία είναι ενεργή σημαίνει ότι είτε ο τρέχων είτε τουλάχιστον ένας από τους προηγούμενους δρομολογητές είναι σε συμφόρηση. Στη δρομολόγηση EDXY εάν ένας δρομολογητής είναι ένα hop μακριά από τον προορισμό σε μια σειρά (ή μια στήλη), η σημαία συμφόρησης σε μία γραμμή (ή σε μια στήλη) χρησιμοποιείται στην απόφαση της δρομολόγησης. Στο παράδειγμα που απεικονίζεται στην εικόνα 2.5 το καλώδιο συμφόρησης στη σειρά προορισμού διαβιβάζεται στο δρομολογητή (1, 2) από το δρομολογητή (1, 1). Επομένως το πακέτο θα δρομολογηθεί στο δρομολογητή (2, 2).



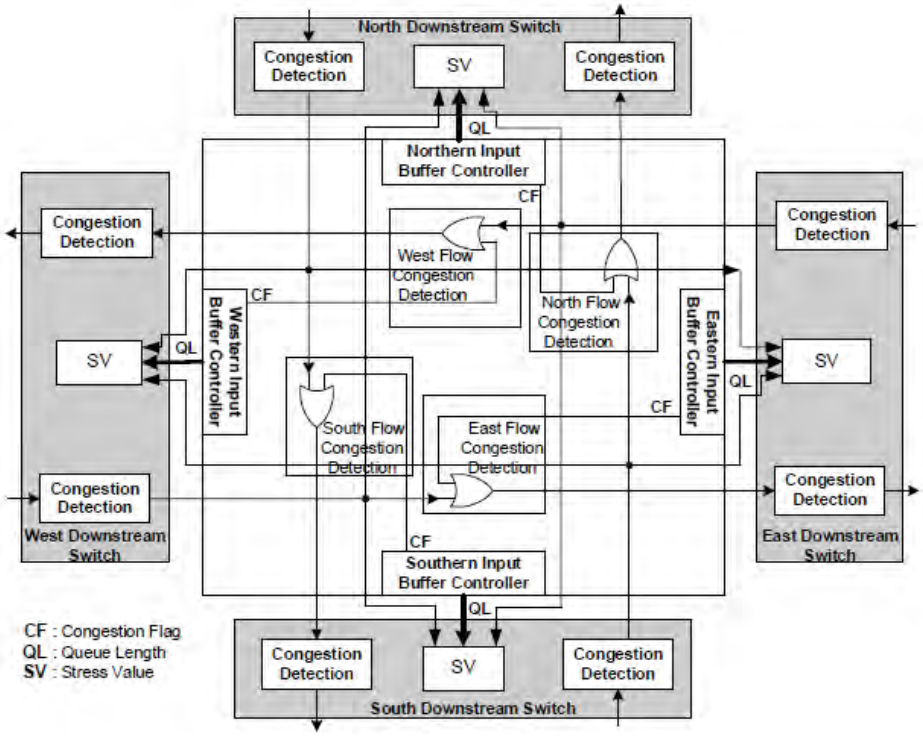
Εικόνα 2.5 Απλό σύστημα NoC με αρχιτεκτονική mesh

Στο EDXY, κάθε δρομολογητής εξετάζει πρώτα τη διεύθυνση προορισμού του ληφθέντος πακέτου. Εάν η διεύθυνση του δρομολογητή δεν είναι ένα hop μακριά από εκείνο του προορισμού είτε στην κατεύθυνση X είτε στην κατεύθυνση Y, ο αλγόριθμος EDXY αγνοεί τη σημαία συμφόρησης και δρομολογεί το πακέτο ακριβώς όπως ο αλγόριθμος DyXY. Ωστόσο αν η διεύθυνση προορισμού είναι μόνο ένα βήμα μακριά από τον δρομολογητή είτε στην κατεύθυνση X είτε στην Y ένα από τα καλώδια συμφόρησης (με βάση τη θέση προορισμού) χρησιμοποιείται επίσης για τη δρομολόγηση. Δηλαδή η τιμή της συμφόρησης χρησιμοποιείται ως το πιο σημαντικό bit της τιμή τάσης. Σε αυτές τις περιπτώσεις παραπέμπουμε στη θύρα που είναι ένα hop από τον προορισμό μας στις κατευθύνσεις X ή Y χαρακτηρίζεται ως κρίσιμη θύρα ενώ η άλλη θύρα χαρακτηρίζεται ως μη κρίσιμη. Για τη μη κρίσιμη θύρα, αγνοούμε την τιμή της σημαίας συμφόρησης και χρησιμοποιείται η τιμή μηδέν για να ευνοηθεί η συγκεκριμένη θύρα έναντι οποιαδήποτε άλλης.

Πρέπει να σημειωθεί ότι τα καλώδια συμφόρησης ενδέχεται να μεταφέρουν ψευδείς πληροφορίες επειδή δεν έχουν πληροφορίες σχετικά με την

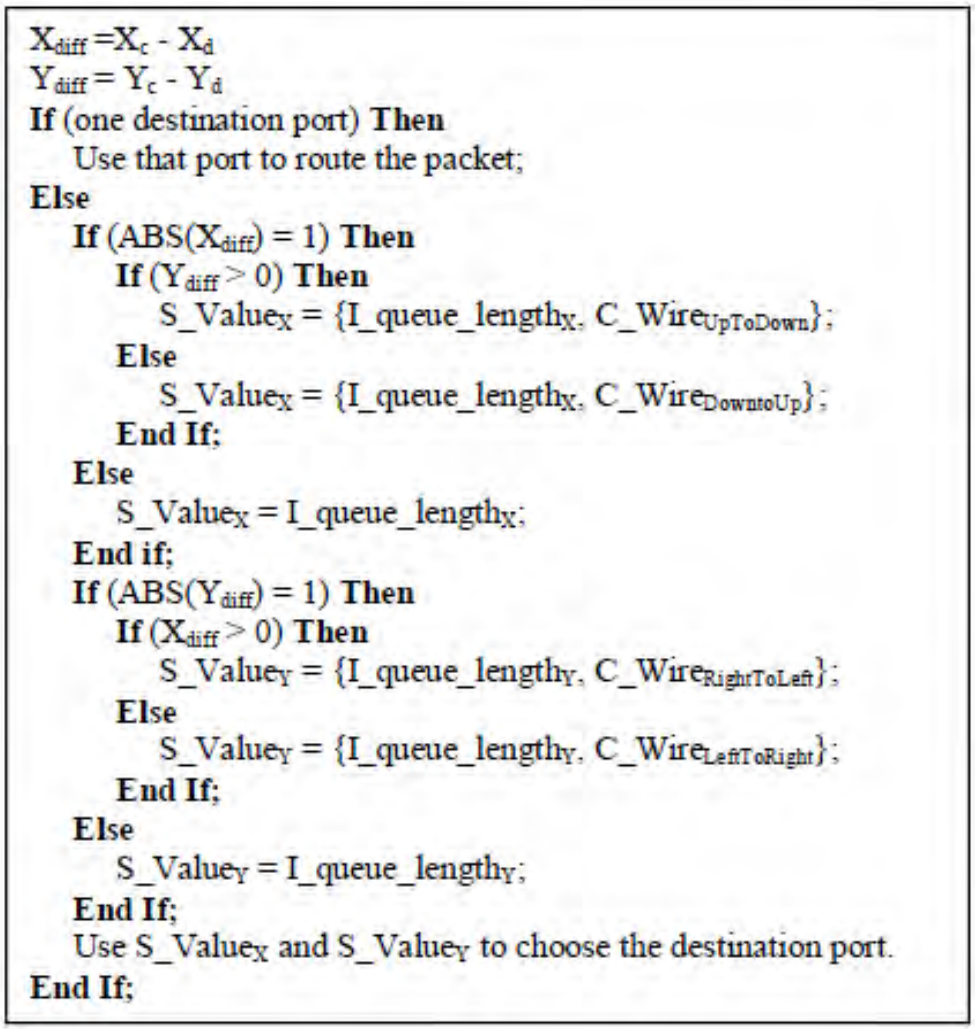
τοποθεσία του κόμβου συμφόρησης σε μια γραμμή (ή στήλη). Συγκρίνουμε τα καλώδια κοινής συμφόρησης με τη γενική περίπτωση στην οποία η θέση του κόμβου συμφόρησης μεταδίδεται επίσης σε μια σειρά (ή στήλη). Τα αποτελέσματα δείχνουν πολύ μικρές διαφορές μεταξύ αυτών των δύο μηχανισμών. Από τη στιγμή που τα αποτελέσματα είναι παρεμφερή προτείνονται να χρησιμοποιηθούν κοινά καλώδια συμφόρησης μιας και έχουν μικρότερη καθυστέρηση στη διάδοση του σήματος. Στις περιπτώσεις που το καλώδιο συμφόρησης πρέπει να χρησιμοποιηθεί στη διαδικασία λήψης αποφάσεων, μπορεί να χρησιμοποιηθεί μια συνάρτηση σχετική με τα χαρακτηριστικά του καλωδίου συμφόρησης για την επιστροφή της τιμής τάσης. Προκειμένου να απλοποιηθεί η υλοποίηση χρησιμοποιήσαμε το καλώδιο συμφόρησης ως MSB του μήκους ουράς για να σχηματίσουμε την τιμή της τάσης.

Για την εφαρμογή του EDXY πρέπει να προστεθεί ένα ακόμη υλικό στον δρομολογητή DyXY. Αυτός ο πρόσθετος εξοπλισμός χωρίζεται σε δύο μέρη. Μια πρόσθετη μονάδα σε κάθε δρομολογητή για να οδηγήσει τα σήματα συμφόρησης σε τέσσερεις κατευθύνσεις. Επιπλέον, κάθε δρομολογητής πρέπει να έχει λογική για να χρησιμοποιεί τα σήματα συμφόρησης στις αποφάσεις δρομολόγησης. Σε κάθε κατεύθυνση, το καλώδιο συμφόρησης εξόδου ρυθμίζεται είτε έχει ρυθμιστεί το σήμα συμφόρησης εισόδου λόγω της συμφόρησης στους προηγούμενους δρομολογητές προς αυτήν την κατεύθυνση είτε το buffer του ρυθμιστή εισόδου για τη δρομολόγηση προς αυτή την κατεύθυνση είναι μεγαλύτερο από την τιμή κατωφλίου. Για την εφαρμογή αυτής της λογικής, χρησιμοποιούνται ένας συγκριτής και μια πύλη OR. Ο συγκριτής χρησιμοποιείται για να συγκρίνει την τιμή της συμφόρησης με την προκαθορισμένη τιμή κατωφλίου. Σε περίπτωση υπέρβασης της τιμής κατωφλίου, η έξοδος του συγκριτή γίνεται ίση με τη μονάδα. Όπως φαίνεται στην εικόνα 2.6 το σήμα συμφόρησης εξόδου είναι αποτέλεσμα σύγκρισης της εξόδου του συγκριτή και του σήματος συμφόρησης εξόδου.



Εικόνα 2.6 Εφαρμογή δρομολογητή EDXY

Ο αλγόριθμος δρομολόγησης θα πρέπει επίσης να τροποποιηθεί για να χρησιμοποιήσει τις νέες πληροφορίες στις αποφάσεις δρομολόγησης. Ο αλγόριθμος δρομολόγησης EDXY δίνεται στην εικόνα 2.7.



Εικόνα 2.7 Αλγόριθμος δρομολόγησης EDXY

2.2.5 Σύγκριση unicast routing protocols

Για την αξιολόγηση της αποτελεσματικότητας του αλγορίθμου EDXY γίνεται η σύγκριση με τρεις άλλους αλγόριθμους δρομολόγησης. Αυτοί οι αλγόριθμοι περιλαμβάνουν το μοντέλο XY, Odd-Even (DyAD) και το DyXY. Ένας προσομοιωτής NoC αναπτύχθηκε σε VHDL για να μοντελοποιήσει την κύρια δομή του δικτύου on-chip και πραγματοποιήθηκαν προσομοιώσεις για τον προσδιορισμό του χαρακτηριστικού latency χρόνου για κάθε δίκτυο. Για όλους τους δρομολογητές, το πλάτος δεδομένων ρυθμίστηκε σε 32 bit. Κάθε

εικονικό κανάλι εισόδου είχε ένα buffer (FIFO) με μέγεθος 6 flits. Η τιμή κατωφλίου συμφόρησης (για τη δρομολόγηση EDXY) ορίστηκε στο 4, πράγμα που σημαίνει ότι η κατάσταση συμφόρησης εξετάστηκε όταν χρησιμοποιήθηκαν 4 από τις 6 θυρίδες buffer.

Για τη μέτρηση των επιδόσεων των αλγορίθμων χρησιμοποιήθηκε η καθυστέρηση που ορίζεται ως ο αριθμός των κύκλων μεταξύ της έναρξης μιας μετάδοσης μηνυμάτων που εκπέμπεται από ένα PE και της χρονικής στιγμής κατά την οποία το μήνυμα παραδίδεται πλήρως στον PE προορισμού. Ο request rate ορίζεται ως ο λόγος των επιτυχημένων injections μηνύματος στη διασύνδεση δικτύου σε σχέση με τον συνολικό αριθμό των προσπαθειών injections. Ο προσομοιωτής warmed up για 3.000 κύκλους και στη συνέχεια μετρήθηκε η μέση απόδοση σε άλλους 100.000 κύκλους.

Ο δρομολογητής χρησιμοποιεί την ελάχιστη πλήρως προσαρμοστική τεχνική αποκλεισμού αδιεξόδου VC που παρουσιάζεται λεπτομερώς στο [9]. Χρησιμοποιήθηκαν τέσσερα ίχνη συνθετικών κυκλοφοριακών σημείων συγκριτικής αξιολόγησης, uniform random, hotspot 5% και 10% και SPLASH-2. Ο πίνακας 2.1 δείχνει τη διαμόρφωση του βασικού δικτύου και τις παραλλαγές που χρησιμοποιούνται στις μελέτες.

Characteristics	**Baseline**	**Variations**
Topology	7 × 7 2D Mesh	15 × 15 2D Mesh
Routing	XY, DyXY και EDXY	Odd-Even
Virtual channels/port	2	0
Flit buffers/VC	6	-
Packet length (flits)	9	15
Traffic workload	Transpose, uniform, hotspot	SPLASH-2 traces
Simulated packets/node	3000	-

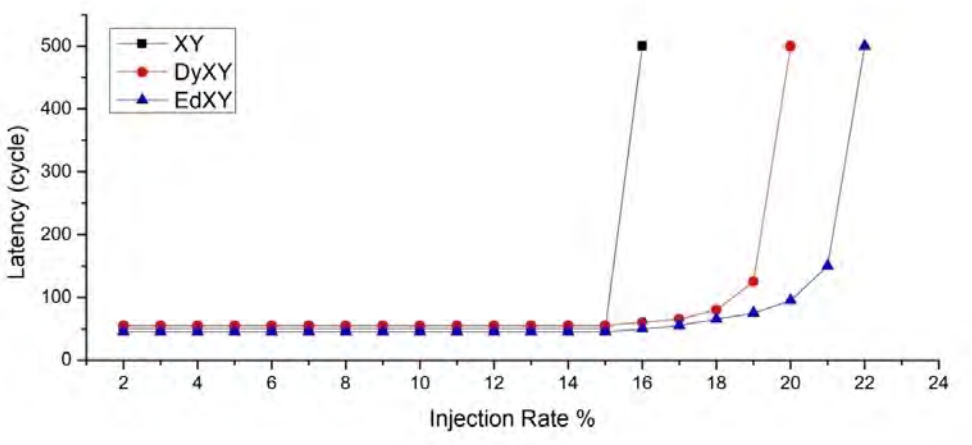
Πίνακας 2.1Διαμόρφωση και παραλλαγές δικτύου προσομοίωσης

2.2.5.1 Πρώτο σετ προσομοίωσης

Στο πρώτο σετ χρησιμοποιήθηκε NoC 7×7 2D τοπολογίας mesh και πακέτα με μήκος 9 flits. Η μέση καθυστέρηση πακέτων για διαφορετικά προφίλ κίνησης παρουσιάζεται στην εικόνα 2.8.

Transpose traffic profile

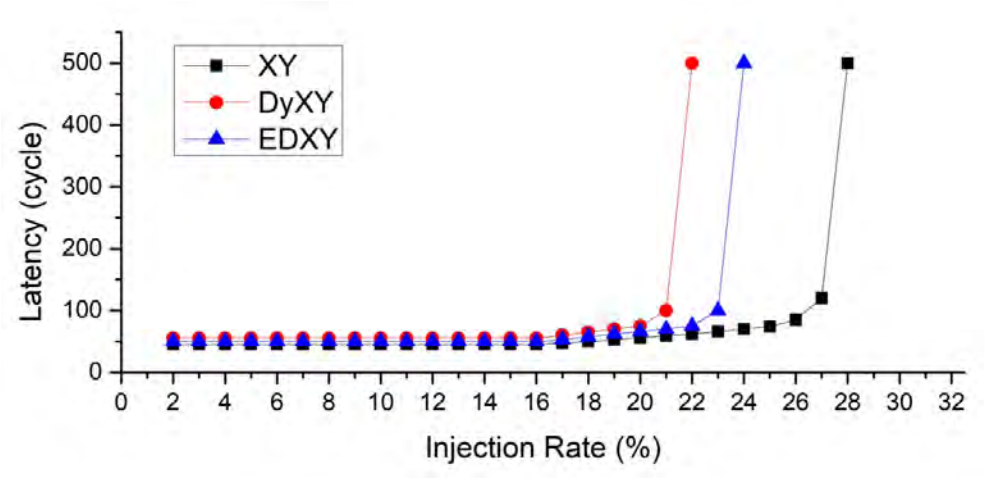
Στο Transpose traffic profile, για ένα δίκτυο $n \times n$ mesh, ένας πυρήνας στη θέση (i, j) $(i, j)(i, j \in [0, n))$ στέλνει μόνο ένα πακέτο δεδομένων σε έναν άλλο πυρήνα στη θέση $(n-1-i, n-1-j)$. Αυτή η μέθοδος κυκλοφορίας της πληροφορίας είναι παρόμοια με την έννοια της μεταφοράς μιας matrix [10]. Σε αυτές τις προσομοιώσεις, κάθε πυρήνας παράγει πακέτα και τα εγχέει στο δίκτυο χρησιμοποιώντας τα χρονικά διαστήματα που προσδιορίζονται χρησιμοποιώντας την εκθετική κατανομή. Αυτό το προφίλ κυκλοφορίας οδηγεί σε μη ομοιόμορφη κατανομή της κυκλοφορίας με μεγάλη κίνηση για τους κεντρικούς κόμβους του πλέγματος. Ως εκ τούτου, μπορεί να δημιουργηθεί κοντά στο κέντρο των σημείων πρόσβασης στο δίκτυο. Όπως φαίνεται στην εικόνα 2.8, εάν ο ρυθμός έγχυσης του πακέτου δεδομένων είναι πολύ χαμηλός, δεν δημιουργούνται hotspots και το σχήμα δρομολόγησης συμπεριφέρεται πανομοιότυπα. Καθώς αυξάνεται ο ρυθμός έγχυσης και δημιουργείται συμφόρηση στο πλέγμα, ο αλγόριθμος EDXY οδηγεί σε μικρότερες μέσες καθυστερήσεις.



Εικόνα 8 Καθυστέρηση έναντι ρυθμού έγχυσης πακέτων για EDXY, DyXY και XY για 7×7 πλέγμα 2D για πακέτα 9-flit με εικονικό κανάλι σε transpose traffic

Uniform traffic profile

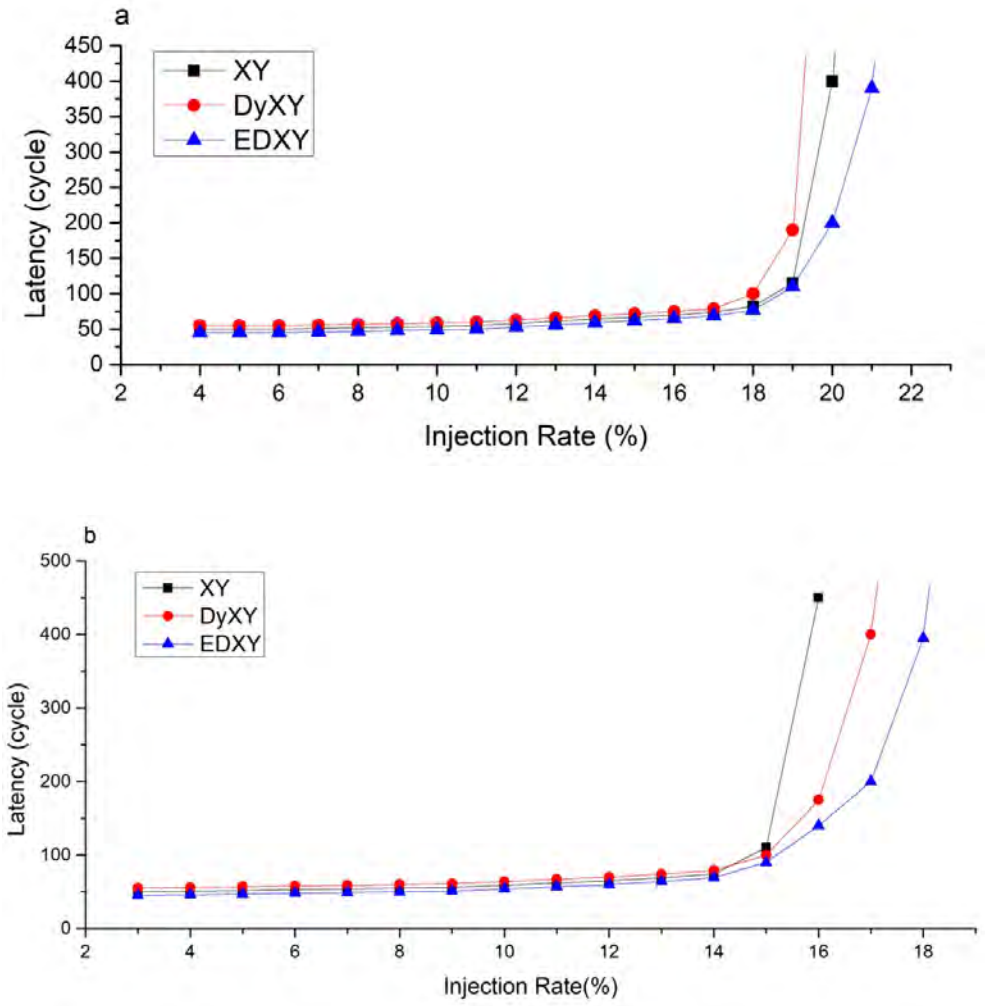
Στο προφίλ Uniform traffic , κάθε κόμβος αποστέλλει μηνύματα σε άλλους κόμβους του δικτύου όπου χρησιμοποιείται ομοιόμορφη κατανομή για την κατασκευή του συνόλου προορισμών κάθε μηνύματος [5][10]. Η μέση καθυστέρηση επικοινωνίας ως συνάρτηση του μέσου ρυθμού έγχυσης μηνύματος φαίνεται στην εικόνα 2.9. Στην περίπτωση αυτή, το σχήμα XY οδηγεί σε χαμηλότερες λανθάνουσες περιόδους, διότι η ομοιόμορφη κίνηση εξισορροπείται με δρομολόγηση XY [11].



Εικόνα 2.9 Καθυστέρηση έναντι ρυθμού έγχυσης πακέτων για EDXY, DyXY και XY για 7×7 πλέγμα 2D για πακέτα 9-flit με εικονικό κανάλι σε uniform random traffic

Hotspot traffic profiles

Ακριβώς όπως και στο uniform traffic profile, κάθε κόμβος στέλνει αρκετά μηνύματα σε άλλους κόμβους του δικτύου ενώ οι κόμβοι (2, 2), (2, 4), (4, 2) και (4, 4) λαμβάνουν 5% και 10% περισσότερα πακέτα στα προφίλ hotspot 5% και 10% αντιστοίχως. Οι μέσες καθυστερήσεις επικοινωνίας ως συνάρτηση του μέσου ρυθμού έγχυσης πακέτων για προφίλ κίνησης σημείου ζεύξης 5% και 10% φαίνονται στην εικόνα 2.10 α) και β) αντίστοιχα. Όπως παρατηρήθηκε από αυτά τα στοιχεία, το EDXY έχει μικρότερη καθυστέρηση σε σύγκριση με άλλα συστήματα και για τα δύο ποσοστά hotspot. Έτσι το EDXY μπορεί να κατανείμει αποτελεσματικότερα την κίνηση μεταξύ των ελάχιστων διαδρομών.



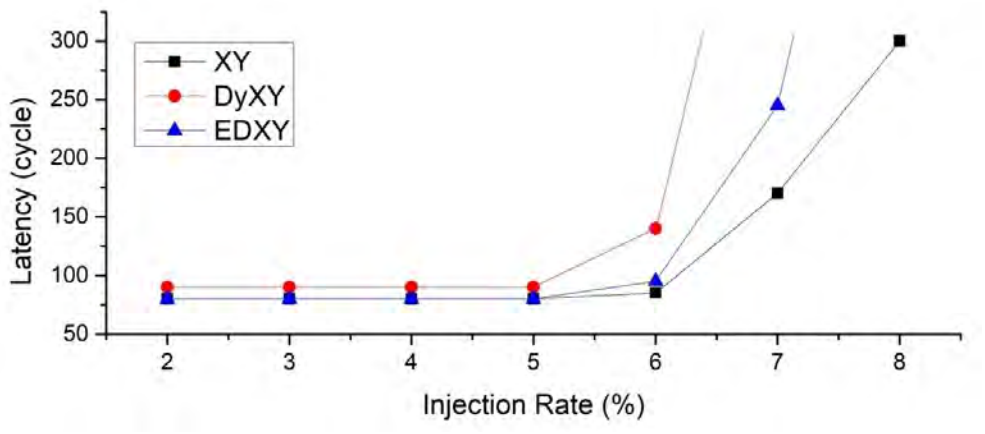
Εικόνα 10 Καθυστέρηση έναντι ρυθμού έγχυσης πακέτων για EDXY, DyXY και XY για 7×7 πλέγμα 2D για πακέτα 9-flit με εικονικό κανάλι για α) hotspot 5% και β) hotspot 10%.

2.2.5.2 Δεύτερο σετ προσομοίωσης

Σε αυτή την ενότητα, αλλάζουμε μερικές από τις παραμέτρους NoC που εξετάζονται στην πρώτη σειρά προσομοίωσης.

Η εικόνα 2.11 δείχνει την καθυστέρηση έναντι του ποσοστού έγχυσης πακέτων για ένα 15×15 mesh NoC κάτω από το προφίλ κίνησης μεταφοράς με πακέτα 9-flit. Για αυτό το μεγάλο δίκτυο, οι προσαρμοστικές προσεγγίσεις

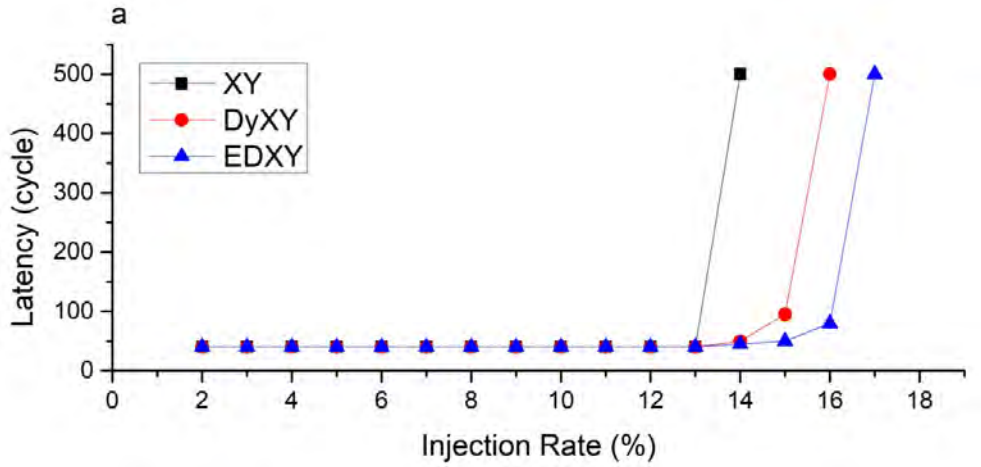
δεν είναι ικανές να εκτελέσουν εκτελούν ούτε τον XY. Δηλαδή, όταν το μέγεθος του δικτύου αυξάνεται, το αποτέλεσμα των πληροφοριών συμφόρησης μειώνεται.

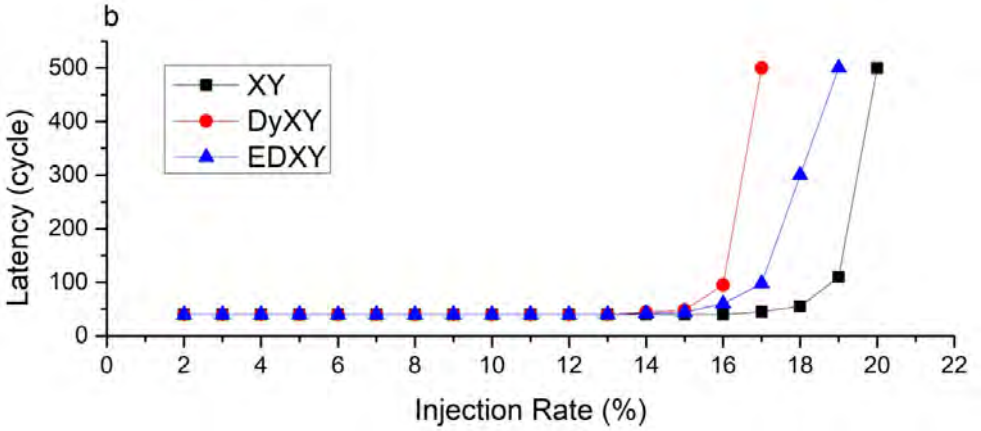


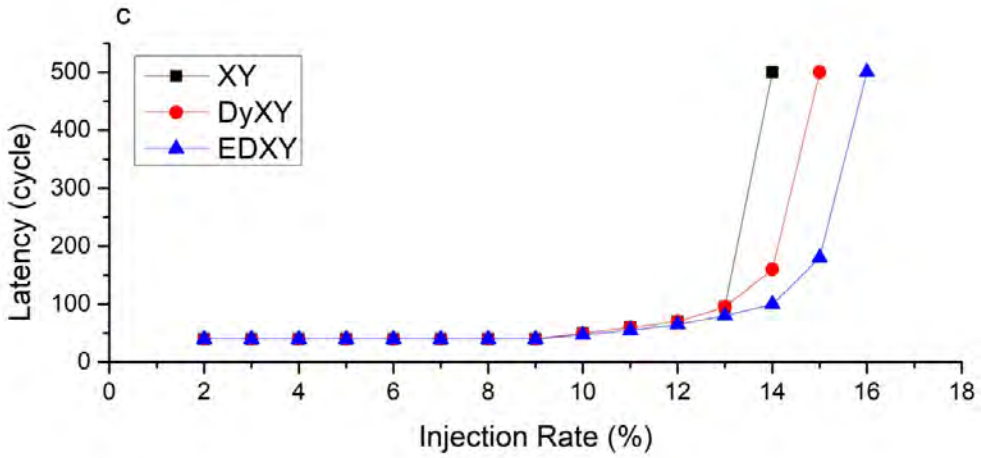
Εικόνα 2.11 Καθυστέρηση έναντι ρυθμού έγχυσης πακέτων για 15×15 mesh με εικονικό κανάλι υπό μεταφορά προφίλ κυκλοφορίας χρησιμοποιώντας πακέτα 9-flit.

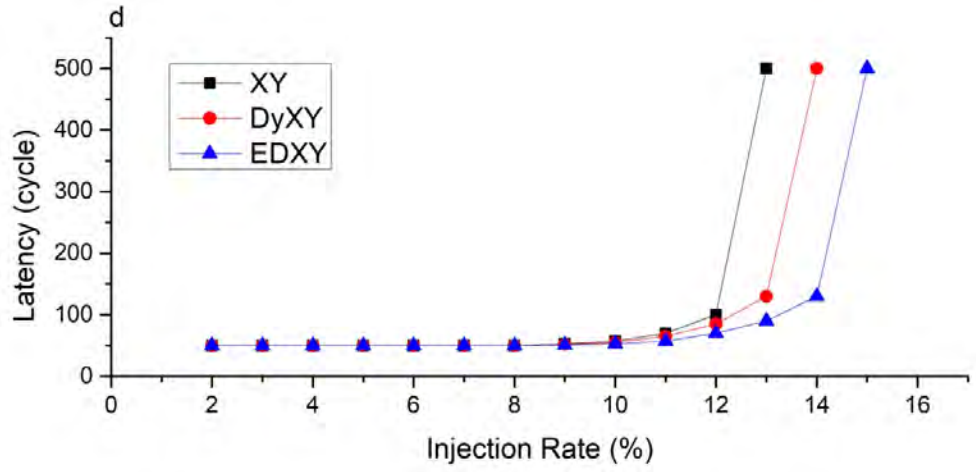
Packet length

Η εικόνα 2.12 δείχνει την καθυστέρηση για τα πακέτα μεγάλης διάρκειας (15 flits) σε ένα πλέγμα 7×7 2D σε διαφορετικά προφίλ κίνησης. Όπως φαίνεται, οι μέσες λανθάνουσες καθυστερήσεις όλων των συστημάτων δρομολόγησης είναι υψηλότερες από άλλα συστήματα. Η αυξημένη μέση καθυστέρηση είναι ένα γνωστό χαρακτηριστικό της δρομολόγησης σκουληκότρυπας με μεγάλα πακέτα. Ο λόγος είναι ότι για τα μεγάλα πακέτα, υπάρχει ανισορροπία στη χρήση των πόρων, επειδή τα πακέτα διαθέτουν πόρους από πολλούς δρομολογητές. Παρόμοια με την περίπτωση πακέτων 9-flit, το σύστημα EDXY εκτελεί καλύτερα από τα DyXY και XY σε όλα τα μοντέλα κυκλοφορίας εκτός από την uniform traffic.





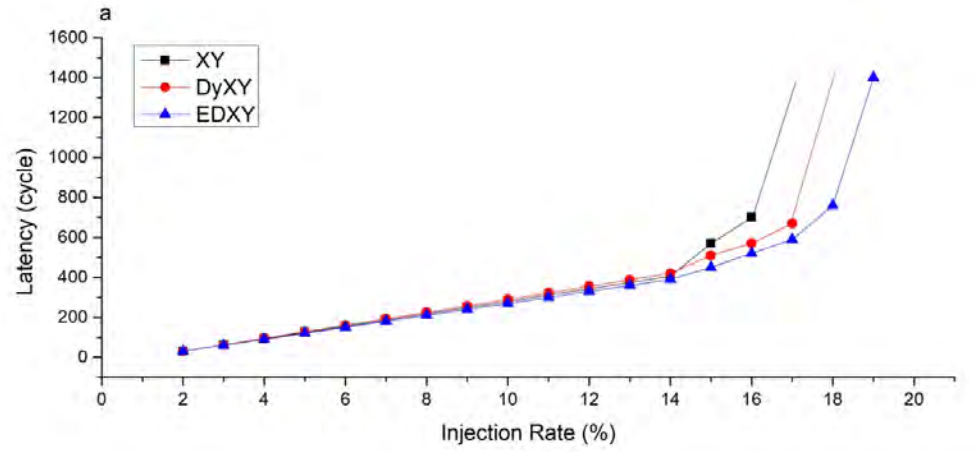


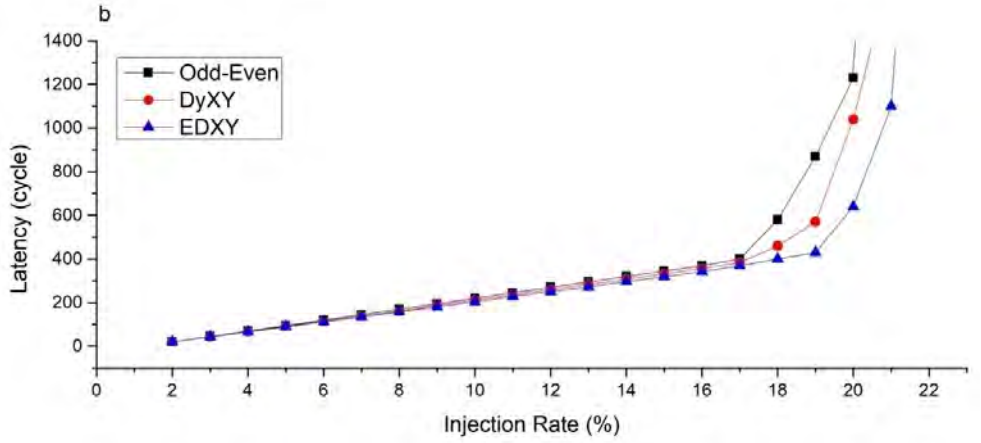


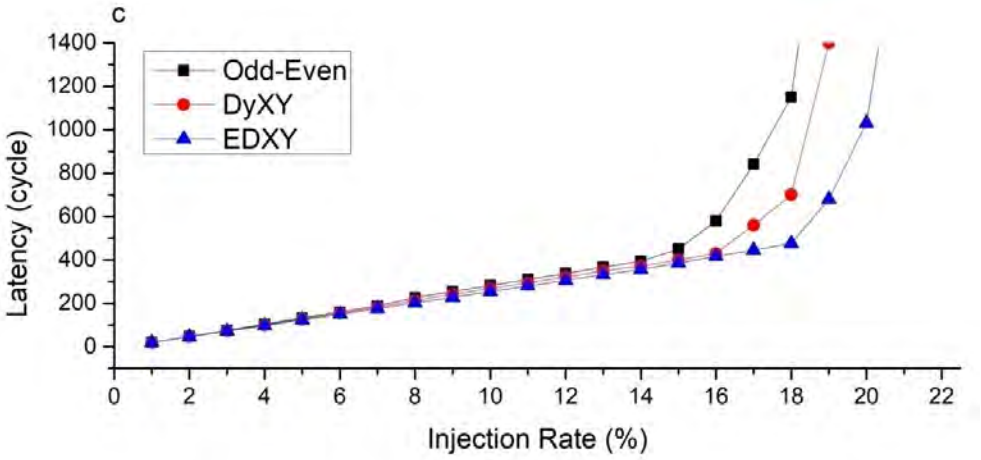
Εικόνα 2.12 Μέση καθυστέρηση έναντι ποσοστού έγχυσης πακέτων σε πλέγμα 7 × 7 2D για πακέτα 15-flit με εικονικό κανάλι α) transpose traffic, β) uniform random traffic, γ) hotspot 5% και δ) hotspot 10%.

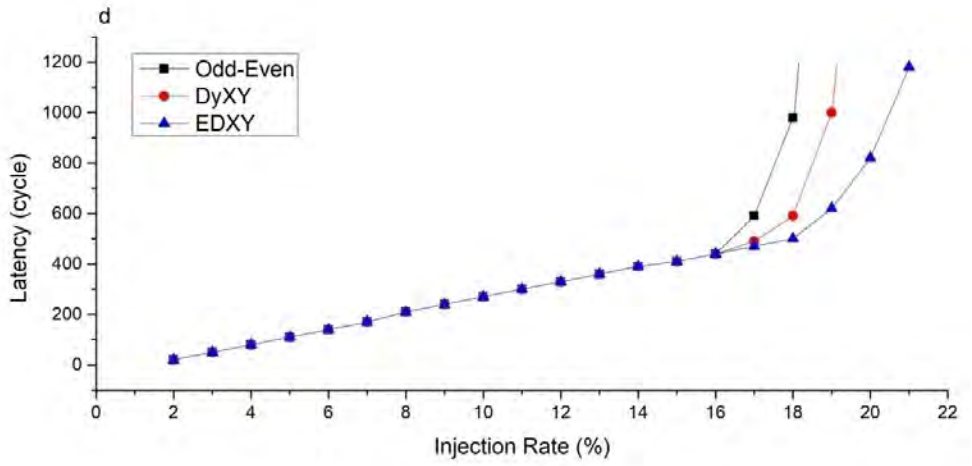
Δίκτυο χωρίς εικονικό κανάλι

Η εικόνα 2.13 δείχνει την καθυστέρηση έναντι του ρυθμού έγχυσης πακέτων για δρομολογητές Odd-Even, DyXY και EDXY χωρίς εικονικά κανάλια. Στην περίπτωση αυτή, θεωρούμε ένα 5×5 2D mesh με πακέτα 5-flit. Το μέγεθος του buffer για κάθε κανάλι είναι 6 flits. Το μοντέλο περιστροφής Odd-Even [8] είναι μια τεχνική για την αποφυγή αδιεξόδου σε NoCs χωρίς εικονικό κανάλι. Για την αποφυγή αδιεξόδου σε DyXY και EDXY, χρησιμοποιείται το μοντέλο περιστροφής Odd-Even. Στην πραγματικότητα, όταν το Odd-Even παρέχει περισσότερες από μία θύρες εξόδου, στον δρομολογητή Odd-Even, επιλέγεται η θύρα στην κατεύθυνση Y ενώ στις δρομολογητές DyXY και EDXY, εξετάζεται η τιμή τάσης για να επιλέξετε μία από τις θύρες. Για αυτές τις προσομοιώσεις, ο πυρήνας (2, 2) λαμβάνει 5% και 10% περισσότερα πακέτα σε προφίλ hotspot 5% και 10%, αντίστοιχα. Όπως δείχνουν τα αποτελέσματα, το EDXY συνεχίζει να αποδίδει καλύτερα από άλλους αλγορίθμους δρομολόγησης σε ένα δίκτυο χωρίς εικονικό κανάλι με προφίλ transpose, uniform, hotspot 5% και hotspot 10%.









Εικόνα 2.13 Καθυστέρηση έναντι ρυθμού έγχυσης πακέτων σε πλέγμα 5×5 2D χωρίς εικονικά κανάλια α) transpose traffic, β) uniform random traffic, γ) hotspot 5%, και δ) hotspot 10%.

2.2.5.3 Σύγκριση υλικού

Για να αξιολογηθεί η γενική επιφάνεια του NoC της EDXY, δημιουργήθηκε ένα μοντέλο σε VHDL συντέθηκε με το Synopsys Design Compiler χρησιμοποιώντας μια τυπική βιβλιοθήκη CMOS. Για όλους τους δρομολογητές, το πλάτος των δεδομένων ορίστηκε σε 32 bits (μέγεθος flit) και κάθε κανάλι είχε δύο εικονικά κανάλια με ένα μέγεθος buffer 6 πτερυγίων. Προκειμένου να επιτευχθούν καλύτερα χαρακτηριστικά απόδοσης ισχύος, τα FIFOs εφαρμόστηκαν με τη χρήση μητρώων.

Στον πίνακα 2.2 φαίνονται οι περιοχές που καλύπτουν οι δρομολογητές μας (στην ίδια αρχιτεκτονική βασισμένη σε διαφορετικό αλγόριθμο). Όπως προκύπτει από τον πίνακα, η περιοχή του EDXY σε σύγκριση με το XY (DyXY) είναι 3,6% (1,5%) μικρότερη.

	XY	DyXY	EDXY
Area (μm^2)	86,107	87,881	89,281

Πίνακας 2.2 Σύγκριση περιοχής XY, DyXY, και EDXY.

2.2.5.4 Power consumption

Η κατανάλωση ισχύος των αλγορίθμων δρομολόγησης EDXY και DyXY υπολογίστηκαν και συγκρίθηκαν κάτω από το uniform traffic χρησιμοποιώντας τον προσομοιωτή Synopsys Prime Power. Κάθε πυρήνας στο NoC παρήγαγε πακέτα βασισμένα στο uniform traffic με τον ίδιο μέσο ρυθμό έγχυσης flits. Το ρολόι του NoC ρυθμίστηκε στο 1 GHz. Δεδομένου ότι η προσομοίωση μετά την σύνθεση είναι πολύ αργή, ο δρομολογητής (3, 3) που βρίσκεται κοντά στο κέντρο ενός πλέγματος 7×7 2D ρυθμίστηκε ως διακόπτης, ενώ για τους άλλους δρομολογητές χρησιμοποιήθηκαν τα μοντέλα RTL. Τα αποτελέσματα για την κατανάλωση ενέργειας παρουσιάζονται στον Πίνακα 2.3. Όπως παρατηρείται από αυτόν τον πίνακα, οι καταναλώσεις ισχύος και των δύο συστημάτων είναι περίπου ίδιες.

	DyXY	EDXY
Κατανάλωση ενέργειας (mW)	27.3	27.7

Πίνακας 2.3 Κατανάλωση ενέργειας του DyXY και του EDXY δρομολογητή σε uniform traffic profile (mW)

2.3 Multicast Routing Protocols

Η multicast επικοινωνία έχει αξιοποιηθεί σε κυρίως σε multicomputers. Οι αλγόριθμοι δρομολόγησης πολυεκπομπής μπορούν να ταξινομηθούν ως α) tree-based και β) path-based.

2.3.1 Unicast-based Multicast Routing

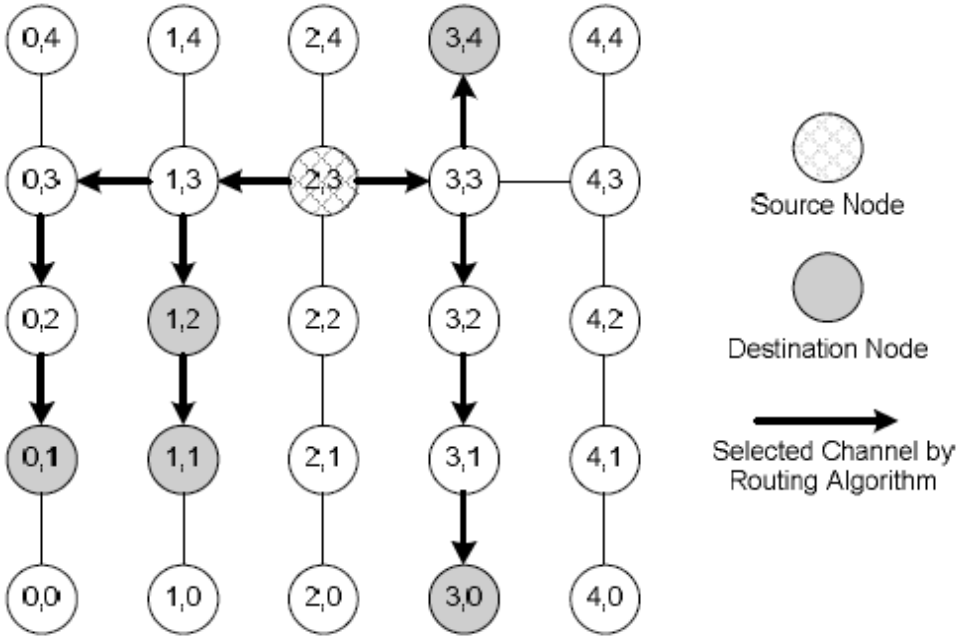
Ο αλγόριθμος Unicast-Based (UB) είναι ένας απλός αλγόριθμος δρομολόγησης πολλαπλών διαδρομών όπου πολλαπλά αντίγραφα του ίδιου μηνύματος, όπως ένα μήνυμα unicast, δρομολογούνται ανεξάρτητα προς κάθε προορισμό ή σε ένα υποσύνολο προορισμών [14]. Το μειονέκτημα αυτού του σχεδίου είναι ότι πολλαπλά αντίγραφα του ίδιου μηνύματος εισάγονται στο δίκτυο, αυξάνοντας την κυκλοφορία του δικτύου. Επιπλέον, κάθε αντίγραφο του μηνύματος έχει σημαντική καθυστέρηση εκκίνησης από την πηγή.

2.3.2 Tree-based Multicast Routing

Στην προσέγγιση δρομολόγησης πολλαπλών δρομολογίων tree-based το πακέτο προορισμού χωρίζεται στην πηγή και αποστέλλονται ξεχωριστά αντίγραφα του μηνύματος μέσω ενός ή περισσοτέρων εξερχομένων καναλιών. Σε αυτή την προσέγγιση δρομολόγησης κατασκευάζουμε ένα spanning tree όπου η πηγή είναι η ρίζα και το μήνυμα αποστέλλεται στο δέντρο [15]. Με αυτόν τον τρόπο, ένα μήνυμα μπορεί να αναπαραχθεί σε μερικούς από τους ενδιάμεσους κόμβους και να προωθηθεί κατά μήκος των πολλαπλών εξερχόμενων καναλιών προς διακεκριμένα υποσύνολα προορισμών. Δεδομένου ότι δεν υπάρχει buffer για μηνύματα σε δρομολογητές, εάν αποκλειστεί ένας κλάδος του δέντρου, αποκλείονται όλα τα υποσύνολα του δέντρου από εκεί και έπειτα [16]. Εάν το μήνυμα δεν μπορεί να προσχωρήσει στο δίκτυό μας, ενδέχεται κάποια κανάλια να είναι σε κατάσταση lockstep για εκτεταμένες χρονικές περιόδους με αποτέλεσμα την υπερφόρτωση του δικτύου. Αν και το tree-based multicast σύστημα μπορεί να χρησιμοποιηθεί αποτελεσματικά σε δίκτυα που χρησιμοποιούν τις τεχνικές store-and-forward και cut-through switching, παρόλα αυτά, προκαλεί υψηλή συμφόρηση στα δίκτυα με wormhole [17]. Ένας tree-based αλγόριθμος δρομολόγησης πολυεκπομπής στα συστήματα NoC ονομάζεται Virtual Circuit Tree Multicasting (VCTM) [18]. Χρησιμοποιώντας τον Virtual Circuit

Table (VCT) και τη Content Addressable Memory (CAM), και στέλνοντας ξεχωριστά μηνύματα ρύθμισης unicast για κάθε προορισμό, χτίζουμε αρκετά δέντρα εικονικών κυκλωμάτων στέλνουμε πολλά μηνύματα πολυεκπομπής στο δίκτυο. Η μέθοδος αυτή, παρόλα αυτά, έχει κάποιες ελλείψεις. Πρώτον, η πολυπλοκότητά της και, κατά συνέπεια, η καθυστέρηση μετάδοσης εξαρτάται σε μεγάλο βαθμό από το μέγεθος του δικτύου. Δεύτερον, ο VCTM είναι ένας αποτελεσματικός αλγόριθμος, κυρίως για συνθήκες δικτύου με χαμηλό ρυθμό εισαγωγής δεδομένων, ενώ για λόγους υψηλής ταχύτητας εισροής (ή φόρτου εργασίας κοντά στον κορεσμό), οι αλγόριθμοι path-based είναι πιο αποδοτικοί. Τρίτον, για την ενημέρωση του VCT, θα πρέπει να αποστέλλονται ξεχωριστά μηνύματα εγκατάστασης unicast ανά προορισμό από τον κόμβο προέλευσης. Εάν αυξηθεί ο αριθμός των προορισμών, ο αριθμός των μηνυμάτων ρύθμισης της unicast θα αυξηθεί, μειώνοντας έτσι την απόδοση. Επομένως, το σύστημα VCTM είναι πιο αποτελεσματικό για εφαρμογές που χρησιμοποιούν ένα μικρό ποσοστό multicast. Τέλος η tree-based multicast επικοινωνία μπορεί να επιτρέψει στα μηνύματα να κρατήσουν πολλά κανάλια ανοιχτά για παρατεταμένες περιόδους και ως εκ τούτου να μειώσουν την απόδοση του δικτύου. Ωστόσο με αυτή την προσέγγιση αποφεύγονται κυκλικές διαδρομές πακέτων.

Ένα παράδειγμα tree-based multicast σε 2-D mesh απεικονίζεται στην εικόνα 2.14 όπου ο κόμβος προέλευσης (2, 3) επιλέγεται ως ρίζα και δημιουργεί ένα δέντρο που εκτείνεται σε σχέση με αυτόν τον κόμβο.



Εικόνα 2.14 Παράδειγμα tree-based multicast αλγορίθμου 5×5 σε 2D-mesh.

Όταν τα flits εισέρχονται στους δρομολογητές στα σημεία διακλάδωσης (κόμβοι (1, 3) και (3, 3)), αντιγράφονται και προωθούνται σε πολλαπλά κανάλια εξόδου. Δεδομένου ότι δεν υπάρχουν μηνύματα προσωρινής αποθήκευσης μηνυμάτων στους δρομολογητές, αν αποκλειστεί ένας κλάδος του δέντρου, όλα τα μηνύματα μπλοκάρονται.

2.3.3 Hamiltonian Path-based Multicast αλγόριθμος δρομολόγησης

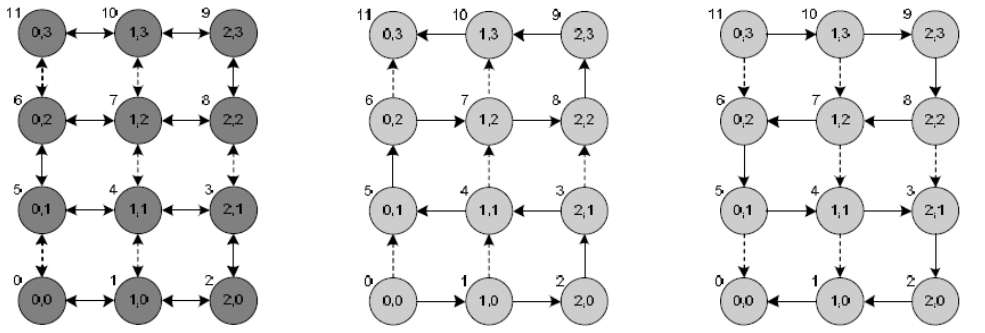
Για να ξεπεραστούν τα μειονεκτήματα των tree-based αλγορίθμων, μπορεί κάποιος να χρησιμοποιήσει αλγορίθμους path-based multicast wormhole. Σε αυτή τη μέθοδο, ένας κόμβος προέλευσης προετοιμάζει ένα μήνυμα για παράδοση σε ένα σύνολο προορισμών, ταξινομώντας πρώτα τις διευθύνσεις του προορισμού με τη σειρά με την οποία πρόκειται να παραδοθούν και στη συνέχεια τοποθετώντας αυτή τη λίστα ταξινόμησης στην κεφαλίδα του μηνύματος. Όταν η κεφαλίδα εισέρχεται σε δρομολογητή με τη διεύθυνση A, ο δρομολογητής ελέγχει αν το A είναι η επόμενη διεύθυνση στην κεφαλίδα.

Αν ναι, η διεύθυνση A αφαιρείται από την επικεφαλίδα του μηνύματος και ένα αντίγραφο των multicast δεδομένων θα παραδοθεί στον τοπικό δρομολογητή και τα flits προωθούνται στον επόμενο κόμβο της διαδρομής. Διαφορετικά, το μήνυμα προωθείται μόνο στον επόμενο κόμβο της διαδρομής. Με αυτό τον τρόπο, το μήνυμα τελικά παραδίδεται σε κάθε προορισμό στην κεφαλίδα [19] [20].

Οι path-based αλγόριθμοι δρομολόγησης βασίζονται στη διαδρομή Hamiltonian . Μια Hamiltonian διαδρομή επισκέπτεται κάθε κόμβο σε ένα γράφημα ακριβώς μία φορά [21]. Για κάθε κόμβο σε μία $m \times n$ αρχιτεκτονική mesh ορίζεται μια ετικέτα $L(x, y)$.

$$L(x,y) = \begin{cases} y \times n + x & \alpha \nu \ \tau o \ n \ \epsilon \acute{\iota} \nu \alpha \iota \ \pi \epsilon \rho \rho \iota \tau \tau \acute{o} \varsigma \\ y \times n + n - x - 1 & \alpha \nu \ \tau o \ n \ \epsilon \acute{\iota} \nu \alpha \iota \ \acute{\alpha} \rho \tau \iota o \varsigma \end{cases}$$

Όπου x, y οι συντεταγμένες ενός κόμβου



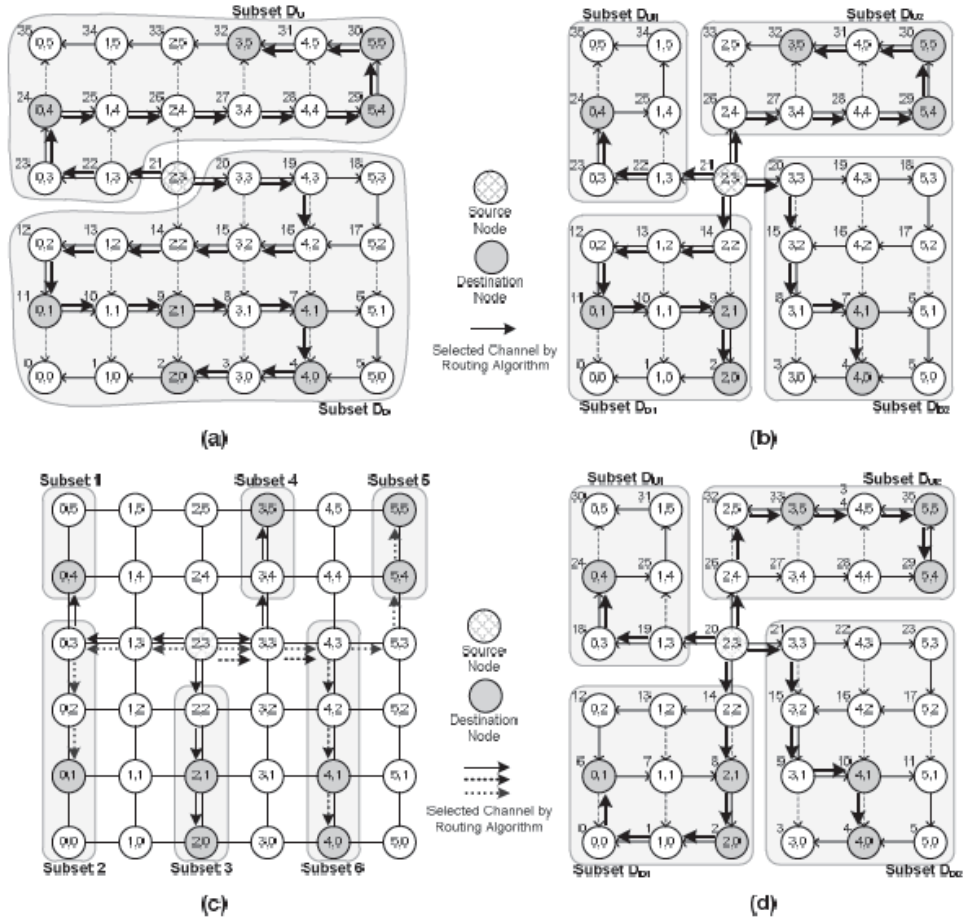
Εικόνα 2.15 α) 3X4 mesh with label assignment και το αντίστοιχο με β) up channel και γ) down channel. Οι συμπαγείς γραμμές δείχνουν το μονοπάτι Hamiltonian και οι διακεκομμένες γραμμές τους συνδέσμους που θα μπορούσαν να χρησιμοποιηθούν για τη μείωση του μήκους διαδρομής στη δρομολόγηση.

Όπως φαίνεται και στην εικόνα 2.15 κατασκευάζονται δύο διαδρομές Hamiltonian (ή υποδίκτυα) με επισήμανση κόμβων [12]. Το δευτερεύον δίκτυο καναλιών (HU) ξεκινά από το (0, 0) ενώ το δευτερεύον δίκτυο (HD) λήγει στο (0, 0). Εάν η ετικέτα του κόμβου προορισμού είναι μεγαλύτερη από την ετικέτα του κόμβου προέλευσης, η δρομολόγηση πραγματοποιείται πάντα στο υποδίκτυο του HU. Διαφορετικά, λαμβάνει χώρα στο υποδίκτυο δικτύου HD. Οι προορισμοί χωρίζονται σε δύο ομάδες. Μία ομάδα περιέχει όλους τους προορισμούς που θα μπορούσαν να προσεγγιστούν χρησιμοποιώντας το υποδίκτυο του HU και ο άλλος περιέχει τους υπόλοιπους προορισμούς στους οποίους θα μπορούσε να επιτευχθεί χρήση του υποδικτύου HD. Για να μειώσουμε το μήκος της διαδρομής τα κατακόρυφα κανάλια που δεν ανοίκοιν στη διαδρομή Hamiltonian (διακεκομμένες γραμμές εικόνας 9) τα χρησιμοποιούμε μόνο στις κατάλληλες κατευθύνσεις. Στην πραγματικότητα, αν σε ένα αλγόριθμο δρομολόγησης όλα τα πακέτα του υποδίκτυου του καναλιού προς τα κάτω (down channel) ακολουθούν μονοπάτια σε αυστηρά αυξουσα (φθίνουσα) σειρά (είτε στη Hamiltonian διαδρομή είτε όχι), δεν μπορεί να δημιουργηθεί κυκλική διαδρομή, οπότε ο αλγόριθμος αυτός δεν μπορεί να οδηγήσει σε αδιέξοδο (deadlock free).

2.3.3.1 Dual-Path (DP) and Multi-Path (MP) Multicast Routing Algorithms

Στον αλγόριθμο δρομολόγησης Dual-Path (DP) το σύνολο των κόμβων προορισμού χωρίζεται σε δύο υποσύνολα D_U και D_D. Κάθε κόμβος σε D_U έχει υψηλότερη ετικέτα από εκείνη του κόμβου προέλευσης και κάθε κόμβος σε D_D έχει χαμηλότερη ετικέτα από εκείνη του κόμβου προέλευσης. Τα D_U και D_D ταξινομούνται στη συνέχεια είτε κατά αύξουσα σειρά είτε κατά φθίνουσα , αντίστοιχα, καθώς η ετικέτα του κάθε κόμβου χρησιμοποιείται ως κλειδί για τη ταξινόμηση. Έτσι, τα μηνύματα πολυεκπομπής από τον κόμβο προέλευσης θα σταλούν στους κόμβους προορισμού σε D_U χρησιμοποιώντας

το υποδίκτυο του H_U και στους κόμβους προορισμού στο D_D χρησιμοποιώντας το υποδίκτυο του H_D.



Εικόνα 2.16 α) Dual-path (DP), β) Multi-path (MP), γ) Column-Path (CP), και δ) Low-Distance (LD) multicast routing από (2, 3). Οι αχρησιμοποίητοι κόμβοι δεν εμφανίζονται.

Στο παράδειγμα της εικόνας 2.16.α βλέπουμε ένα mesh δίκτυο 6X6 όπου ο κόμβος (2, 3) θα στείλει multicast μηνύματα στους κόμβους (2, 0), (4, 0), (0, 1), (2, 1), (4, 1) (5, 4), (3, 5) και (5, 5). Δημιουργούνται δύο υποσύνολα. Το πρώτο υποσύνολο (D_U), το οποίο περιέχει όλους τους προορισμούς που θα μπορούσαν να προσεγγιστούν από τον κόμβο προέλευσης χρησιμοποιώντας υποσύστημα H_U, και περιλαμβάνει τους κόμβους (0, 4), (5, 4), (5, 5) και (3,

5). Το δεύτερο υποσύνολο (D_D), το οποίο έχει τους όλους τους υπόλοιπους προορισμούς που θα μπορούσαν να προσεγγιστούν χρησιμοποιώντας το υποδίκτυο του H_D, περιλαμβάνει τους κόμβους (2, 0), (4, 0), (4, 1), (2, 1) (0, 1). Ορισμένοι από τους κατακόρυφους συνδέσμους που δεν ανήκουν στις διαδρομές Hamiltonian χρησιμοποιούνται κατάλληλα, για την ελαχιστοποίηση των διαδρομών.

Σε αυτό το παράδειγμα, καθώς οι περισσότεροι κόμβοι έχουν τέσσερα κανάλια εξόδου στο πλέγμα 2D, μπορούν να χρησιμοποιηθούν έως και τέσσερις ανεξάρτητες διαδρομές για την παράδοση ενός μηνύματος. Επομένως, τα υποσύνολα προορισμού διπλής διαδρομής των D_U και D_D χωρίζονται επίσης. Το υποσύνολο D_U χωρίζεται σε δύο υποσύνολα. Το ένα αποτελείται από τους κόμβους των οποίων οι συντεταγμένες x είναι μεγαλύτερες ή ίσες με εκείνες της πηγής και το άλλο υποσύνολο περιέχει τους υπόλοιπους D_U κόμβους. Το σύνολο D_D χωρίζεται με παρόμοιο τρόπο. Ως εκ τούτου, όλοι οι προορισμοί του μηνύματος πολλαπλών μεταδόσεων ομαδοποιούνται σε τέσσερις διαφορετικές υποσυνθέσεις έτσι ώστε όλοι οι προορισμοί σε ένα υποσύνολο να βρίσκονται σε ένα από τα τέσσερα τεταρτημόρια. Για το παράδειγμα πολλαπλών διαδρομών που φαίνεται στην εικόνα 2.16.β η πηγή μας βρίσκεται στο (2, 3) και το σύνολο θέσεων κατανέμεται αρχικά σε δύο υποσύνολα το D_U = {(0, 4), (5, 4), (3, 5), (5, 5)} και το D_D = {(2, 0) 0), (4, 1), (2, 1), (0,1)}. Από την εικόνα 9 α) βλέπουμε ότι το D_U χωρίζεται σε δύο υποσύνολα D_{U1} = {(0, 4)} και D_{U2} = {(5, 4), (3, 5), (5, 5)}. Με τον ίδιο τρόπο, η D_D διαιρείται σε δύο υποσύνολα το D_{D1} = {(0, 1), (2, 1), (2, 0)} και το D_{D2} = {(4, 0), (4, 1)}. Η διπλή διαδρομή και η πολλαπλή διαδρομή είναι deadlock free και θα μπορούσαν να χρησιμοποιηθούν ταυτόχρονα για τη δρομολόγηση unicast και multicast.

2.3.3.2 Column-Path (CP) Multicast Routing

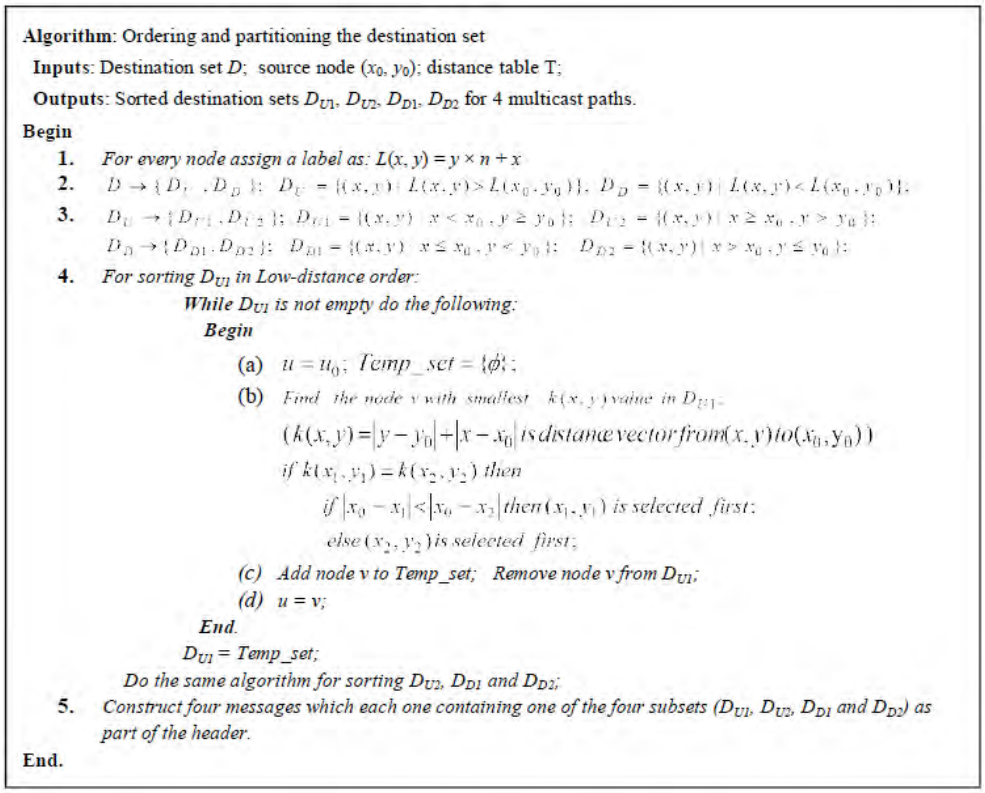
Σε αυτήν τη μέθοδο, το σύνολο κόμβων προορισμού χωρίζεται σε υποσύνολα 2k όπου k είναι ο αριθμός των στηλών στο πλέγμα. Επίσης σε αυτή τη μέθοδο, το πολύ δύο μηνύματα θα αντιγραφούν σε κάθε στήλη. Εάν μια στήλη του πλέγματος έχει έναν ή περισσότερους προορισμούς στις σειρές πάνω από αυτήν της πηγής, τότε αποστέλλεται ένα αντίγραφο του μηνύματος σε όλους τους κόμβους που βρίσκονται στις σειρές πάνω από αυτήν της πηγής. Ομοίως, εάν μια στήλη έχει έναν ή περισσότερους προορισμούς στις γραμμές κάτω από εκείνη της πηγής, τότε αποστέλλεται ένα άλλο αντίγραφο του μηνύματος σε όλους τους κόμβους που βρίσκονται στις γραμμές κάτω από αυτήν της πηγής. Ένα αντίγραφο του μηνύματος αποστέλλεται σε μια στήλη, εάν όλοι οι προορισμοί της στήλης αυτής βρίσκονται κάτω ή πάνω από τον κόμβο προέλευσης. Στην εικόνα 2.16.γ φαίνεται ένα παράδειγμα όπου ένα multicast μήνυμα στέλνεται στους κόμβους (2, 0), (4, 0), (0, 1), (2, 1), (4, 1), (0, 4), (5, 4), (3, 5) και (5, 5) από τον κόμβο προέλευσης (2, 3) χρησιμοποιώντας τον αλγόριθμο δρομολόγησης διαδρομής (CP). Έξι αντίγραφα του μηνύματος χρησιμοποιούνται για την επίτευξη της επιθυμητής λειτουργίας πολλαπλής διανομής. Ο αλγόριθμος δρομολόγησης που χρησιμοποιείται από αυτό το σχήμα βασίζεται στον αλγόριθμο δρομολόγησης XY, ο οποίος είναι deadlock free. Ωστόσο, δεδομένου ότι η δρομολόγηση CP, παρόμοια με τη μέθοδο δρομολόγησης βασισμένη σε unicast, παράγει πάρα πολλά μηνύματα (δηλ. Το πολύ 2k αντίγραφα του μηνύματος στη δρομολόγηση CP και το πολύ 4 και 2 στις γραμμές MP και DP, αντίστοιχα). Έχει σχετικά μεγάλες καθυστερήσεις στα τελευταία μηνύματα μιας και λόγω της αποστολής πολλαπλών μηνυμάτων έχουμε καθυστέρηση στην εκκίνηση αποστολής. Επιπλέον, επειδή πολλά μηνύματα πολλαπλής διανομής αποστέλλονται μέσω των στηλών από κάθε κόμβο προέλευσης, η απόδοση του δικτύου υποβαθμίζεται.

2.3.3.3 Low-Distance (LD) Path-based Multicast Routing

Τρία είναι τα βασικά χαρακτηριστικά του αλγορίθμου 3 Low-Distance (LD) Path-based Multicast.

α) Χρησιμοποιεί μια διαμέριση δικτύου παρόμοια με την τεχνική δρομολόγησης multi-path multicast όπου μπορούν να σχηματιστούν έως και τέσσερις ομάδες προορισμού.

β) Η ταξινόμηση των προορισμών στη διαδρομή θα πρέπει να βελτιστοποιηθεί για να μειωθεί η απόσταση της διαδρομής πολλαπλών διαδρομών. Αυτό επιτυγχάνεται με την προσθήκη κώδικα που βελτιώνει την απόδοση του αλγορίθμου σε σύγκριση με εκείνες προηγούμενων αλγορίθμων δρομολόγησης πολλαπλών διαδρομών με βάση τη διαδρομή. Ο αντίστοιχος αλγόριθμος δρομολόγησης απεικονίζεται στην εικόνα 2.17.



Εικόνα 2.17 Κεφαλίδα μηνύματος για δρομολόγηση πολλαπλών διαδρομών Low Distance (LD).

Σε αυτόν τον αλγόριθμο, για κάθε κόμβο αποδίδεται μια ετικέτα που δίνεται από τον τύπο $L(x, y) = y \times n + x$. Παρόμοια με τον αλγόριθμο multicast πολλαπλών διαδρομών, το σύνολο κόμβων προορισμού χωρίζεται σε τέσσερα υποσύνολα των D_{U1}, D_{U2}, D_{D1} και D_{D2}. Τα υποσύνολα τότε ταξινομούνται στη σειρά Low-Distance με το διάνυσμα απόστασης κάθε κόμβου να χρησιμοποιείται ως κλειδί για τη διαλογή. Το διάνυσμα απόστασης κάθε κόμβου υπολογίζεται από $k = |y - y_0| + |x - x_0|$. Για να ταξινομήσουμε τους προορισμούς με τη σειρά Low-Distance , πρώτα ο κόμβος (v), που έχει το διάνυσμα χαμηλότερης απόστασης στον κόμβο προέλευσης (u_0), τοποθετείται στο Temp_set και αφαιρείται από το υποσύνολο. Στη συνέχεια, ο επιλεγμένος κόμβος θα θεωρηθεί ως κόμβος προέλευσης. Αν το αρχικό υποσύνολο δεν

είναι άδειο, αυτή η ακολουθία θα επαναληφθεί. Διαφορετικά, το Temp_set που περιέχει το ταξινομημένο υποσύνολο προορισμού τοποθετείται στο αρχικό υποσύνολο. Αν υπάρχουν δύο κόμβοι με ίσο διανυσματικό απόστασης σε σύγκριση με τον κόμβο προέλευσης, θα επιλεγεί πρώτα αυτός με τη μικρότερη διάσταση x σε σχέση με αυτόν του κόμβου προέλευσης. Στη συνέχεια, το αρχικό υποσύνολο θα τοποθετηθεί στην κεφαλίδα του μηνύματος.

γ) Για τη δρομολόγηση των μηνυμάτων στους προορισμούς, ο αλγόριθμος χρησιμοποιεί το μοντέλο Odd-Even turn [6] [8]. Το μοντέλο Odd-Even turn απαγορεύει τη μεταφορά δεδομένων από ανατολικά προς βόρεια και ανατολικά προς νότια (βορρά προς δύση και από νότια προς δυτικά) σε όλους τους δρομολογητές που βρίσκονται σε μια άρτια (περιττή) στήλη. Αυτό καθιστά την τεχνική ως έναν προσαρμοστικό αλγόριθμο χωρίς αδιέξοδο που χρησιμοποιεί τη συντομότερη διαδρομή. Δεδομένου ότι είναι χωρίς αδιέξοδο, δεν υπάρχει ανάγκη για την εφαρμογή εικονικών καναλιών στο δρομολογητή για την αποφυγή αδιεξόδου. Η προσθήκη εικονικών καναλιών έχει επίπτωση στην ταχύτητα δεδομένου ότι η πολυπλοκότητα και η λανθάνουσα κατάσταση του ελεγκτή αυξάνεται με τον αριθμό των εικονικών καναλιών λόγω των αυξημένων απαιτήσεων ρύθμισης και διαιτησίας [22].

Σε ορισμένες περιπτώσεις για τη δρομολόγηση ενός μηνύματος πολλαπλής μετάδοσης από έναν προορισμό στον επόμενο μέσω μιας ελάχιστης διαδρομής, απαιτείται η διάδοση δεδομένων μέσω απαγορευμένης στροφής. Για να αποφευχθεί ένα πιθανό αδιέξοδο σε αυτές τις περιπτώσεις, το μήνυμα αντιγράφεται πρώτα από τον πρώτο προορισμό και στη συνέχεια ένα αντίγραφο του μηνύματος θα μεταδοθεί ξανά στην επόμενη διεύθυνση προορισμού μέσω των καναλιών κατανάλωσης.

2.3.4 Hamiltonian Adaptive Multicast Unicast Method (HAMUM)

Έχουν προταθεί διάφοροι αλγόριθμοι δρομολόγησης πολλαπλών διαδρομών που βασίζονται στη διαδρομή Hamiltonian για να εγγυηθούν το μη ενδεχόμενο αδιεξόδου [17] [19]. Αλλά οι παραδοσιακοί αλγόριθμοι με βάση τη διαδρομή είναι καθοριστικοί τόσο για την κυκλοφορία unicast όσο και για την multicast, η οποία υποβαθμίζει σημαντικά την απόδοση. Αυτό ήταν και το κίνητρο να αναζητηθούν νέες μέθοδοι με βάση τη διαδρομή για να φέρουμε την προσαρμοστικότητα τόσο για την κυκλοφορία unicast όσο και για την multicast χωρίς τη χρησιμοποίηση εικονικών καναλιών. Για να βελτιωθεί η pathbased μέθοδος, παρουσιάζεται μια προσαρμοστική μέθοδος χωρίς αδιέξοδο για την προσαρμογή σε όλα τα μοντέλα με βάση το Hamiltonian. Στην πραγματικότητα, σε αντίθεση με άλλα προσαρμοστικά μοντέλα σε δίκτυα επικοινωνιών που ισχύουν μόνο για κίνηση unicast, η προτεινόμενη μέθοδος χειρίζεται ταυτόχρονα τόσο την κίνηση unicast όσο και την multicast.

Στα παραδοσιακά μοντέλα δρομολόγησης με βάση τη διαδρομή όπως MP και CP, τόσο για μηνύματα unicast όσο και για τα multicast σύμφωνα με την τρέχουσα και την επόμενη θέση των κόμβων προορισμού, χρησιμοποιείται μόνο ένας συντομότερος δρόμος από τον αλγόριθμο δρομολόγησης. Επομένως, η απόδοση του δικτύου υποβαθμίζεται χρησιμοποιώντας αυτούς τους αλγορίθμους δρομολόγησης. Το HAMUM μπορεί να λύσει αυτό το πρόβλημα στους αλγόριθμους δρομολόγησης με βάση τη διαδρομή, κατευθύνοντας προσαρμοστικά και τα μηνύματα unicast και multicast μέσω ενδιάμεσων προορισμών.

Στη μέθοδο αυτή, οι θέσεις στις οποίες μπορούν να ληφθούν ορισμένες κατευθύνσεις είναι περιορισμένες, για να αποφευχθεί το αδιέξοδο. Οι κανόνες που ρυθμίζουν το προτεινόμενο σχήμα κατηγοριοποιούνται στο κατώτερο δίκτυο καναλιών ως εξής:

Όλοι οι κόμβοι στις άρτιες σειρές έχουν κατώτερες ετικέτες από τους γειτονικούς κόμβους τους στις βόρειες και ανατολικές κατευθύνσεις. Ενώ οι κόμβοι στις περιττές σειρές έχουν χαμηλότερες ετικέτες από τους γείτονές τους σε βόρειες και δυτικές κατευθύνσεις. Επομένως, για το υποδίκτυο:

Κανόνας 1: Επιτρέπονται βόρειες και ανατολικές κατευθύνσεις στις άρτιες σειρές.

Κανόνας 2: Οι κατευθύνσεις Βορρά και Δύσης επιτρέπονται στις περιττές σειρές.

Ομοίως, όλοι οι κόμβοι στις άρτιες σειρές έχουν υψηλότερες ετικέτες από τους γειτονικούς τους κόμβους στη νότια και δυτική κατεύθυνση. Ενώ οι κόμβοι στις περιττές σειρές έχουν υψηλότερες ετικέτες από τους γείτονές τους στις νότιες και ανατολικές κατευθύνσεις. Έτσι, για το δευτερεύον δίκτυο καναλιών:

Κανόνας 1: Οι νότιες και οι δυτικές κατευθύνσεις επιτρέπονται σε άρτιες σειρές.

Κανόνας 2: Οι νότιες και οι ανατολικές κατευθύνσεις επιτρέπονται σε περιττές σειρές.

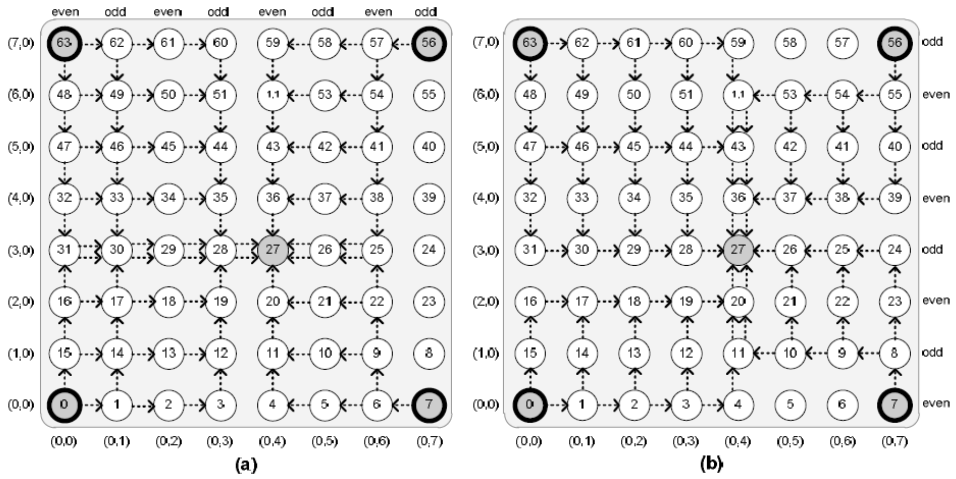
Ένα μήνυμα θα προωθηθεί προς τον προορισμό στη ντετερμινιστική Hamiltonian, όταν ο τρέχων κόμβος εντοπίζεται μία σειρά προς τα νότια (ή βόρεια) της σειράς προορισμού στο υποδίκτυο των καναλιών κατώτερου δικτύου. Η εικόνα 2.18 δείχνει τον ψευδοκώδικα του μοντέλου HAMUM που εκτελείται σε κάθε δρομολογητή όταν φτάσει ένα νέο πακέτο. Εφόσον οι κανόνες διατηρούν τα μηνύματα που μεταδίδονται σε αυστηρά αύξουσα σειρά ή και φθίνουσα σειρά , αποτρέπει την εμφάνιση αδιεξόδου.

```
Algorithm HAMUM is
-- (Cx,Cy) : Current node , (Dx,Dy) : Destination node
Begin
    If (Dy = Cy) then                                   --Current & Dest. are in the same row
        If (Dx = Cx) then                               --Current& Dest. are in the same column
            direction <= Local;                         --Packet sends to the Local direction
        Elsif (Dx > Cx) then                            --Dest. is to the East of the Current
            direction <= East;                          --Dest. is to the West of the Current
        Else  direction <= West;
        End if;
    Elsif (Dy > Cy) then                                -- up channel Subnetwork
        If ( Cy mod 2 = 0 ) then                        --rule1 in the even rows
            If ( Dx > Cx ) and ( Dy - Cy > 1 ) then     --Dest. is in the East & more than 1 row to Current
                direction <= North or East              --North or East direction can be chosen
            Elsif ( Dx > Cx ) and ( Dy - Cy = 1 ) then  --Dest. is in the East & 1 row to the Current
                direction <= East;                      --Packet sends to the East direction
            Else  direction <= North;                   --IF Dest. is in the West of the Current, select North
            End if;
        Elsif ( Cy mod 2 /= 0 ) then                    --rule2 in odd rows
            If ( Dx < Cx ) and ( Dy - Cy > 1 ) then     --Dest. is in the West & more than 1 row to Current
                direction <= North or West              --North or West direction can be chosen
            Elsif ( Dx < Cx ) and ( Dy - Cy = 1 ) then  --Dest. is in the West & 1 row to the Current
                direction <= West;                      --Packet sends to the West direction
            Else  direction <= North;                   --IF Dest. is in the West of the Current, select North
            End if;
        End if;
    Elsif ( Dy < Cy ) then                              -- down channel Subnetwork
        If ( Cy mod 2 = 0 ) then                        --rule1 in even rows
            If ( Dx < Cx ) and ( Cy - Dy > 1 ) then     --Dest. is in the West & more than 1 row to Current
                direction <= South or West              --South or West direction can be chosen
            Elsif ( Dx < Cx ) and ( Cy - Dy = 1 ) then  --Dest. is in the West & 1 row to the Current
                direction <= West;                      --Packet sends to the West direction
            Else  direction <= South;                   --IF Dest. is in the West of the Current, select South
            End If;
        Elsif ( Cy mod 2 /= 0 ) then                    --rule2 in odd rows
            If ( Dx > Cx ) and ( Cy - Dy > 1 ) then     --Dest. is in the East & more than 1 row to Current
                direction <= South or East              --South or East direction can be chosen
            Elsif ( Dx > Cx ) and ( Cy - Dy = 1 )then   --Dest. is in the East & 1 row to the Current
                direction <= East;                      --Packet sends to the East direction
            Else  direction <= South;                   --IF Dest. is in the West of the Current, select South
            End if;
        End if;
    End if;
End HAMUM;
```

Εικόνα 2.18 Ο ψευδοκώδικας του HANUM

2.3.4.1 Unicast Aspect of HAMUM

Βάσει της προτεινόμενης μεθόδου, οποιοσδήποτε ενδιάμεσος κόμβος πρέπει πρώτα να καθορίσει τη σειρά των κατευθύνσεων προς τις οποίες ένα πακέτο μπορεί να προωθηθεί για το επόμενο hop βάσει του κανόνα 1 και του κανόνα 2. Όπως αναφέρθηκε προηγουμένως, σύμφωνα με τις ετικέτες προέλευσης και προορισμού, η δρομολόγηση μπορεί να στο πάνω ή στο κάτω υποδίκτυο του καναλιού.



Εικόνα 2.19 Όλες οι πιθανές ελάχιστες διαδρομές από τους κόμβους προέλευσης 63, 56, 7 και 0 στον κόμβο προορισμού 27 α) στο μοντέλο Odd-Even και β) στην περίπτωση unicast του HAMUM.

Εξετάζουμε την περίπτωση όπου ο προορισμός ενός μηνύματος βρίσκεται στα δυτικά της πηγής στο υποδίκτυο του καναλιού επάνω (π.χ. κόμβος προέλευσης 7 και προορισμός 27 όπως φαίνεται στην εικόνα 2.19. Εάν ο τρέχων κόμβος βρίσκεται σε μία περιττή σειρά, ο δρομολογητής μπορεί να δρομολογήσει το μήνυμα προς τη δυτική ή τη βόρεια κατεύθυνση, λόγω του αλγορίθμου υποδικτύου του Hamiltonian up channel. Εάν ο τρέχων κόμβος βρίσκεται σε μια άρτια σειρά, αρχικά το μήνυμα πρέπει να δρομολογηθεί προς τη βόρεια κατεύθυνση (για να φτάσει σε περιττή σειρά), και στη συνέχεια, θα μπορούσε να δρομολογηθεί μέσω της δυτικής της βόρειας κατεύθυνσης. Στο κατώτερο υποδίκτυο, χρησιμοποιώντας τη διαδρομή Hamiltonian, το πακέτο μπορεί να επιλέξει δυτική ή βόρεια κατεύθυνση στις άρτιες σειρές και ανατολική ή βόρεια κατεύθυνση στις περιττές. Επιπρόσθετα, εάν ο τρέχων κόμβος βρίσκεται μια σειρά στα νότια της σειράς προορισμού στο υποδίκτυο των καναλιών προς τα πάνω, το μήνυμα θα δρομολογηθεί προς τη δυτική ή τη βόρεια κατεύθυνση αν ο τρέχων κόμβος βρίσκεται σε άρτια σειρά και αν ο τρέχων κόμβος είναι σε περιττή σειρά το

πακέτο θα δρομολογηθεί προς τη βόρεια κατεύθυνση. Στην εικόνα 2.19.β, φαίνονται όλες οι πιθανές ελάχιστες διαδρομές δρομολόγησης του HAMUM για τέσσερα μηνύματα σε 8x8 2D-mesh. Τουλάχιστον μία ελάχιστη διαδρομή μπορεί πάντα να επιλεγεί με την προτεινόμενη μέθοδο για οποιοδήποτε ζεύγος πηγής και προορισμού. Δεδομένου ότι το μοντέλο Odd-Even είναι από τα κλασικά μοντέλα αλγορίθμου στη unicast πτυχή του, θα το συγκρίνουμε με την κλασική μορφή του. Όλες οι πιθανές διαδρομές δρομολόγησης για το μοντέλο Odd-Even φαίνονται εικόνα 2.19.α. Για να συγκρίνουμε τους δύο αλγόριθμους μεταξύ τους, χρησιμοποιούμε τον συντελεστή βαθμού προσαρμοστικότητας (DoA) [3] [4], ο οποίος είναι ο αριθμός των ελάχιστων διαδρομών που μπορούν να ληφθούν από ένα μήνυμα για να ταξιδέψουν από έναν κόμβο προέλευσης (Sx, Sy) σε έναν κόμβο προορισμού (Dx, Dy). Υποθέστε ότι το Δx, Δy ορίζεται ως Δx = Dx - Sx και Δy = Dy - Sy και dx = | Δx |, και dy = | Δy |. Ο βαθμός προσαρμοστικότητας για έναν πλήρως προσαρμοστικό αλγόριθμο δίνεται από:

$$DoA(\ fully\ adaptive\ routing)_{s,d} = \frac{(d_x + d_y)!}{d_x!\, d_y!}$$

Με βάση το Hamiltonian Path, μπορούν να υπάρχουν οκτώ διαφορετικές καταστάσεις θέσης σύμφωνα με τη θέση του κόμβου προέλευσης (άρτια ή περιττή γραμμή), τη θέση του κόμβου προορισμού (άρτια ή περιττή) και την κατεύθυνση του κόμβου προορισμού (αριστερή ή δεξιά πλευρά του κόμβος προέλευσης), όπως αυτές συνοψίζονται στον πίνακα 2.4.

State	Source position (odd/even row)	Destination position (odd/even row)	Destination direction (left/right)
1	even	even	right (east)
2	even	odd	right (east)
3	even	even	left (west)
4	even	odd	left (west)
5	odd	even	right (east)
6	odd	odd	right (east)
7	odd	even	left (west)
8	odd	odd	left (west)

Πίνακας 2.4 Οι οκτώ διαφορετικές καταστάσεις θέσης των κόμβων προέλευσης και προορισμού.

Αρχικά, υπολογίζουμε το DoA για όλα τα μηνύματα unicast στο ανώτερο δίκτυο καναλιών, και στη συνέχεια χρησιμοποιούμε παρόμοιο τρόπο για να υπολογίσουμε το DoA για το υποδίκτυο των καναλιών κάτω. Όπως μπορεί να φανεί στην εικόνα 2.20, το DoA της κατάστασης 1 και 8 είναι ίσο και μπορεί να υπολογιστεί ως:

$$DoA(1)_{s,d} = \frac{(d_x + D)!}{d_x!D!} \text{ όπου } D = \left\lfloor \left| \frac{d_y}{2} \right| \right\rfloor$$

Για όλες τις άλλες καταστάσεις, η συνάρτηση DoA υπολογίζεται ως:

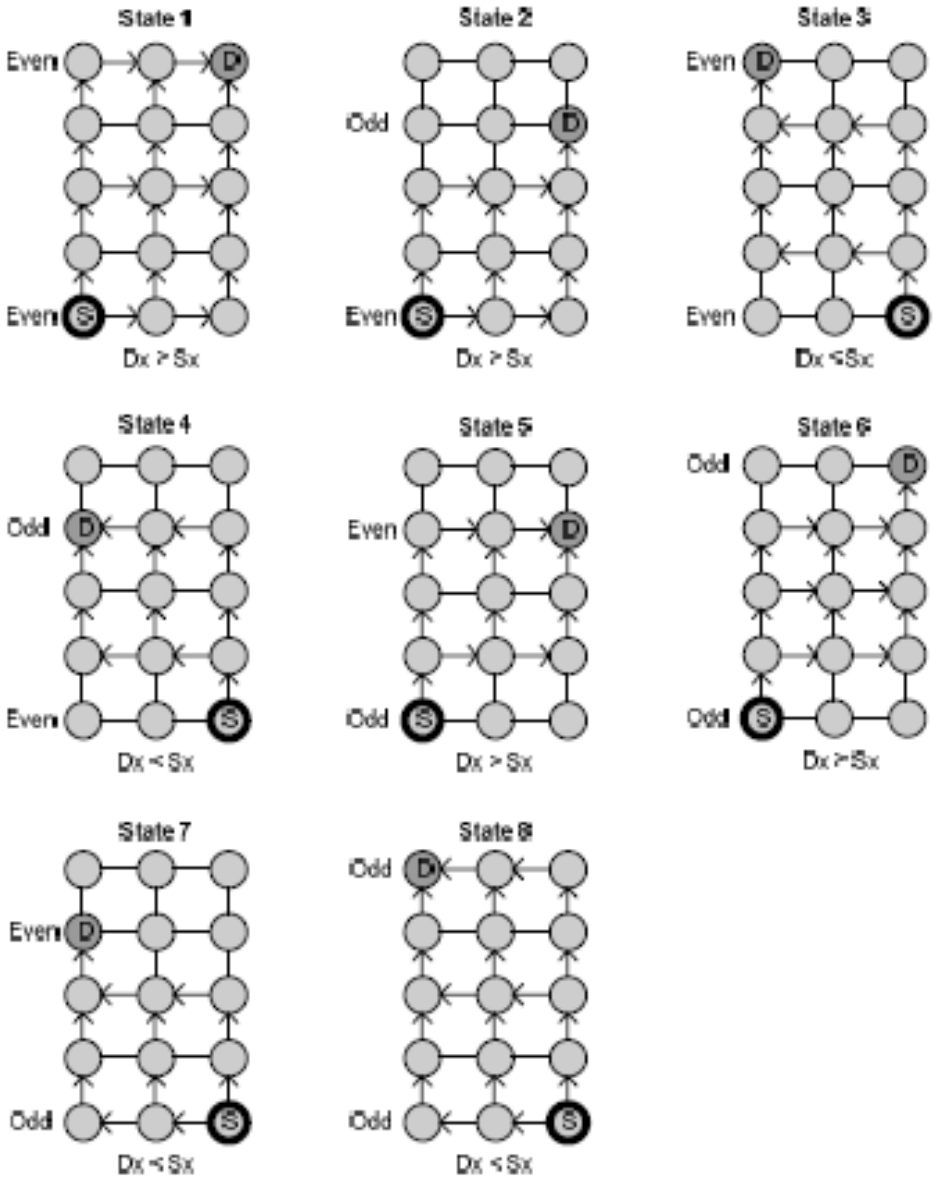
$$DoA(2)_{s,d} = \frac{(d_x + D')!}{d_x!D'!} \text{ όπου } D' = \left\lfloor \left| \frac{d_y - 1}{2} \right| \right\rfloor$$

Αυτές οι εξισώσεις μπορούν να συνοψιστούν ως εξής: DoA του υποδίκτυου καναλιών επάνω:

$$DoA(up-channel)_{s,d} = \begin{cases} DoA(1)_{s,d} & για\ καταστασεις\ 1\ και\ 8 \\ DoA(1)_{s,d}\ για\ όλλες\ τις\ άλλες\ περιπτώσεις \end{cases}$$

DoA του υποδικτύου κάτω καναλιού:

$$DoA\left(down-channel\right)_{s,d}=\begin{cases} DoA\left(1\right)_{s,d} & για\ κατάστασεις\ 3\ και\ 6 \\ DoA\left(1\right)_{s,d}\ για\ όλλες\ τις\ άλλες\ περιπτώσεις \end{cases}$$

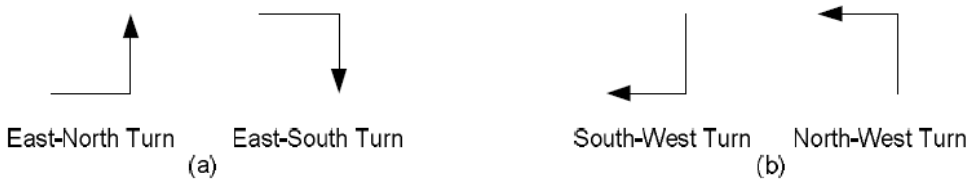


Εικόνα 2.20 Οκτώ διαφορετικές καταστάσεις τοποθεσίας στα υπο-δίκτυο καναλιών.

Το μοντέλο Odd-Even [8] περιορίζει τις θέσεις όπου μπορούν να πραγματοποιηθούν ορισμένοι τύποι στροφών. Ενώ οι κανόνες HAMUM βασίζονται στις σειρές των πλεγμάτων, ενώ οι κανόνες του μοντέλου Odd-Even βασίζονται στις στήλες. Οι κανόνες Odd-Even περιγράφονται ως εξής:

Κανόνας 1: Οι στροφές Ανατολή-Βορράς και Ανατολής-Νότου δεν μπορούν να ληφθούν σε άρτιες στήλες (Εικόνα 2.21.α)

Κανόνας 2: Οι στροφές Βοράς-Δύση και Νότος-Δύση δεν μπορούν να ληφθούν σε περιττές στήλες (Εικόνα 2.21.β).



Εικόνα 2.21 Ο κανόνας του Μοντέλου περιστροφής (Odd-Even): α) απαγορεύεται η περιστροφή στις άρτιες στήλες β) απαγορεύεται η περιστροφή στις περιττές στήλες.

Ο βαθμός προσαρμοστικότητας του μοντέλου Odd-Even Turn υπολογίζεται ως [8]:

Όταν ο κόμβος προορισμού βρίσκεται στη δεξιά πλευρά του κόμβου προέλευσης (Δx> 0):

$$DoA(\Delta x>0)_{s,d}=\begin{cases} DoA(2)_{s,d} \text{ αν η στήλη του κόμβου προέλευσης είναι επιτρεπτή και} \\ \text{ο προορισμός βρίσκεται σε περιττή στήλη} \\ DoA(1)_{s,d} \text{ σε οποιαδήποτε άλλη περίπτωση} \end{cases}$$

Όταν ο κόμβος προορισμού βρίσκεται στην αριστερή πλευρά του κόμβου προέλευσης (Δx <0):

$$DoA(\Delta x<0)_{s,d}=\begin{cases} DoA(1)_{s,d} \ \alpha\nu\ \eta\ \sigma\tau\acute{\eta}\lambda\eta\ \tau\sigma\upsilon\ \kappa\acute{o}\mu\beta\sigma\upsilon\ \pi\rho\sigma\acute{\epsilon}\lambda\epsilon\upsilon\sigma\eta\varsigma\ \epsilon\acute{\iota}\nu\alpha\iota\ \epsilon\pi\iota\tau\rho\epsilon\pi\tau\acute{\eta}\ \kappa\alpha\iota\ \Delta x=0 \\ DoA(1)_{s,d}\ \sigma\epsilon\ o\pi o\iota\alpha\delta\acute{\eta}\pi o\tau\epsilon\ \acute{\alpha}\lambda\lambda\eta\ \pi\epsilon\rho\acute{\iota}\pi\tau\omega\sigma\eta \end{cases}$$

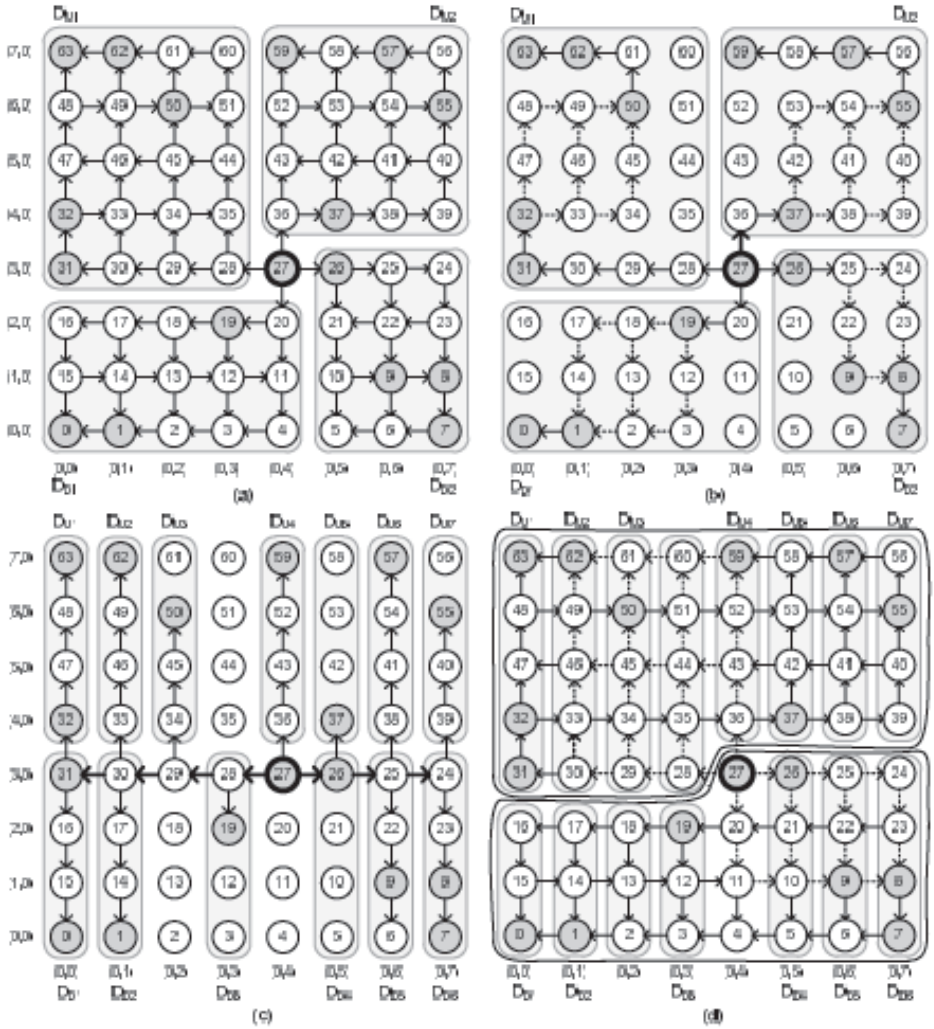
Ο βαθμός προσαρμοστικότητας του HAMUM και του μοντέλου Odd-Even είναι περίπου το ίδιο. Δεδομένου ότι το μοντέλο Odd-Even δεν μπορεί να χρησιμοποιηθεί για την κυκλοφορία πολλαπλών μεταδόσεων, το HAMUM είναι όχι μόνο συμβατό με την κυκλοφορία πολλαπλών μεταδόσεων, αλλά παρέχει και προσαρμοστικότητα τόσο για την κυκλοφορία unicast όσο και για multicast.

2.3.4.2 Multicast Aspect of HAMUM

Η προσαρμοστική μέθοδος επηρεάζει τους αλγορίθμους δρομολόγησης πολλαπλών διαδρομών με βάση τη διαδρομή. Για το σκοπό αυτό, εφαρμόζουμε HAMUM στους αλγόριθμους Multi-Path (MP) και Path-Path (CP).

1) Adaptive Multi-Path (AMP) Routing Algorithm

Η εικόνα 2.22.α δείχνει ένα παράδειγμα MP όπου ο κόμβος πηγής 27 (3, 4) δημιουργεί ένα μήνυμα πολυεκπομπής προς αποστολή προς προορισμούς 31, 9, 59, 8, 50, 57, 26, 19, 62, 37, 0 , 63, 1, 7, 32, 55. Συνεπώς, οργανώνονται δύο υποσύνολα.



Εικόνα 2.22 α) Multi-Path (MP), β) Adaptive Multi-Path (AMP), γ) Column-Path (CP), και δ) Adaptive Column-Path (ACP) αλγόριθμοι δρομολόγησης

Το πρώτο υποσύνολο (D_U) έχει όλους τους προορισμούς με υψηλότερη ετικέτα και το δεύτερο (D_D) έχει τους υπόλοιπους προορισμούς. Στη συνέχεια, το D_U χωρίζεται σε δύο υποσύνολα, D_{U1} = {31, 32, 50, 62, 63} και D_{U2} = {37, 55, 57, 59}. Με τον ίδιο τρόπο, το D_D χωρίζεται σε δύο υποσύνολα, το D_{D1} = {19, 1, 0} και το D_{D2} = {26, 9, 8, 7}. Τέλος, ένα πακέτο ανά υποσύνολο θα πρέπει να δημιουργηθεί και να αποσταλεί από τον κόμβο προέλευσης στο δίκτυο. Όλα τα πακέτα πρέπει να ακολουθούν το

Hamiltonian μονοπάτι και να φτάσουν στους προορισμούς με προκαθορισμένη σειρά.

Το AMP (Adaptive MP) είναι το προσαρμοστικό μοντέλο του αλγόριθμου MP μετά την εφαρμογή του προτεινόμενου προσαρμοστικού μοντέλου στον αλγόριθμο MP. Το παράδειγμα που χρησιμοποιείται για το MP στην εικόνα 16 β), το μήνυμα πολλαπλής μετάδοσης μπορεί να προωθηθεί με τρεις διαφορετικούς τρόπους από τον κόμβο 37 στον κόμβο 55 (32 έως 50, 19 προς 1 και 26 έως 8).

2) Adaptive Column-Path (ACP) Routing Algorithm

Χρησιμοποιούμε το παράδειγμα της εικόνας 2.22.α για CP. Δεκατρία αντίγραφα του μηνύματος χρησιμοποιούνται για την επίτευξη της επιθυμητής λειτουργίας πολλαπλής διανομής Εικόνα 2.22γ. Αν και οι προορισμοί 1 και 62 βρίσκονται στην ίδια στήλη, αποστέλλονται δύο αντίγραφα μηνύματος σε αυτήν τη στήλη, αφού ένας από τους προορισμούς βρίσκεται πάνω από τη σειρά του κόμβου προέλευσης και ο άλλος κάτω.

Το ACP, που ανήκει στο Adaptive CP είναι η προσαρμοστική μέθοδος του αρχικού CP εκμεταλλευόμενος το προσαρμοστικό μοντέλο. Για να υποδείξουμε πώς το προσαρμοστικό σχήμα επηρεάζει τον αλγόριθμο CP, όπως απεικονίζεται στην εικόνα 2.22.δ, και πάλι δεκατρία αντίγραφα του μηνύματος πολλαπλών μεταδόσεων πρέπει να χρησιμοποιούνται για την επίτευξη της επιθυμητής λειτουργίας πολλαπλής διανομής. Αλλά σε αυτό το σχήμα για απλότητα, εξετάζουμε μόνο δύο υποσύνολα το D_{U2} και D_{D6}. Λόγω της χρήσης του προτεινόμενου προσαρμοστικού σχήματος στο CP, κάθε μήνυμα πολυεκπομπής μπορεί να παραδοθεί στο υποσύνολο του μέσω διαφόρων διαδρομών που υποδεικνύονται με διακεκομμένες γραμμές.

2.3.5 Σύγκριτικά αποτελέσματα multicast routing protocols

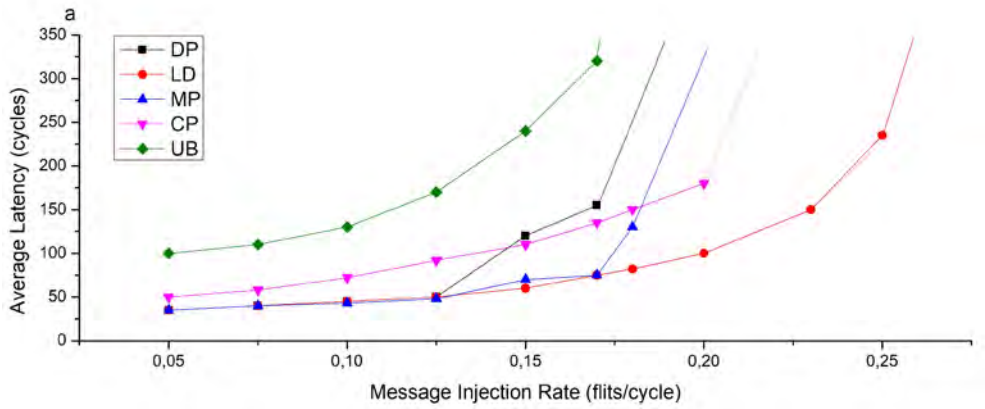
Για να αξιολογηθεί η αποτελεσματικότητα του αλγορίθμου δρομολόγησης πολλαπλών διαδρομών που βασίζεται Low Distance (LD), εφαρμόζονται οι αλγόριθμοι Dual-Path (DP), Multi-Path (MP), Path Column (CP) και UB (Unicast-Based). Οι MP και CP χρησιμοποιούνται επίσης για την αξιολόγηση του μοντέλου δρομολόγησης HAMUM. Έχει αναπτυχθεί για το σκοπό αυτό ένας προσομοιωτής ακριβείας του NoC . Ο προσομοιωτής υπολογίζει τη μέση καθυστέρηση και την κατανάλωση ενέργειας για τη μετάδοση μηνυμάτων. Οι εισόδους προσομοιωτή περιλαμβάνουν το μέγεθος του πίνακα, τη συχνότητα λειτουργίας, τον αλγόριθμο δρομολόγησης, το πλάτος σύνδεσης και τον τύπο κυκλοφορίας. Ο προσομοιωτής μπορεί να δημιουργήσει διαφορετικά προφίλ. Για να υπολογίσουμε την κατανάλωση ενέργειας, χρησιμοποιούμε τις λειτουργίες της βιβλιοθήκης Orion [23]. Για όλους τους δρομολογητές, το πλάτος δεδομένων και η συχνότητα ρυθμίστηκαν στα 32 bit και 1 GHz αντίστοιχα, γεγονός που οδήγησε σε εύρος ζώνης 32 Gb/s. Κάθε κανάλι εισόδου έχει μέγεθος buffer (FIFO) 8 flits με το όριο συμφόρησης που έχει ρυθμιστεί στο 75% της συνολικής χωρητικότητας buffer. Το μέγεθος του μηνύματος υποτίθεται ότι είναι 16 flits. Επιπλέον, υποθέσαμε επίσης ότι η τοπολογία του πλέγματος είναι 2D mesh και οι καθυστερήσεις στα καλώδια δεν θα υπερέβαιναν την περίοδο του ρολογιού. Για τη μέτρηση απόδοσης, χρησιμοποιούμε την καθυστέρηση πολλαπλής διανομής που ορίζεται ως ο αριθμός των κύκλων μεταξύ της έναρξης της λειτουργίας μηνύματος πολλαπλών μεταδόσεων και του χρόνου κατά τον οποίο η ουρά του μηνύματος πολλαπλής κλήσης φτάνει σε όλους τους προορισμούς. Το CP έχει την πιο περίπλοκη διαδικασία για την προετοιμασία των μηνυμάτων πολυεκπομπής, ενώ η DP έχει την ευκολότερη διαδικασία [14]. Ο μηχανισμός προετοιμασίας αποτελείται από το διαχωρισμό του συνόλου προορισμού σε κατάλληλα υποσύνολα και τη δημιουργία πολλαπλών αντιγράφων του μηνύματος. Για τον υπολογισμό του χρόνου προετοιμασίας, έχουν εκτελεστεί

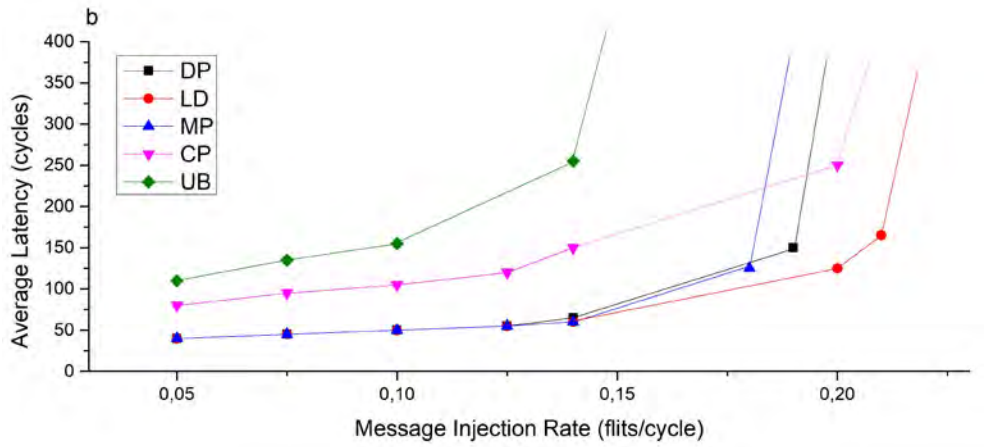
από τον προσομοιωτή διάφορες σειρές προορισμών πολυεκπομπής. Σε αυτά τα σύνολα δοκιμών, ο μέσος χρόνος προετοιμασίας για την ολοκλήρωση των μηνυμάτων multicast στους αλγόριθμους DP, MP, LD και CP ήταν 35, 46, 46 και 82 κύκλοι, αντίστοιχα. Επειδή ο αλγόριθμος DP παράγει μόνο 2 μηνύματα multicast, είναι ο καλύτερος μεταξύ των άλλων αλγορίθμων και το CP είναι το χειρότερο όσον αφορά την καθυστέρηση εκκίνησης.

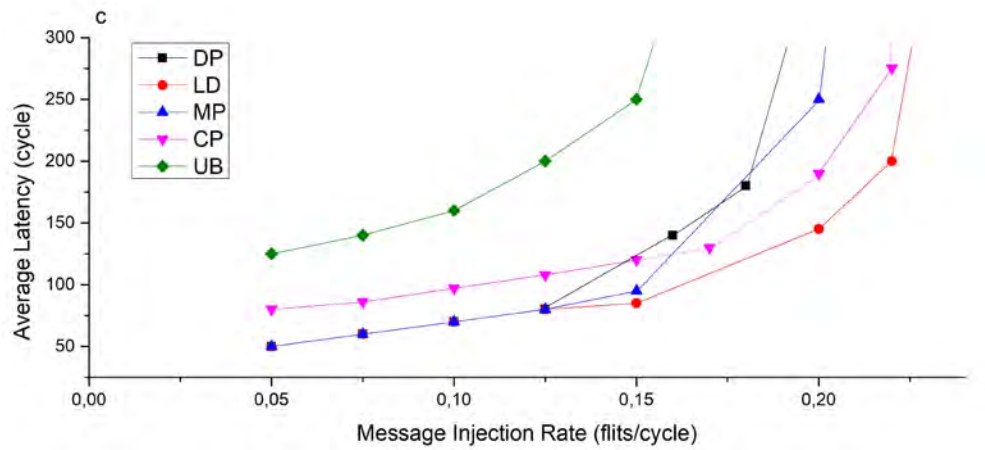
2.3.5.1 Multicast Traffic Profile

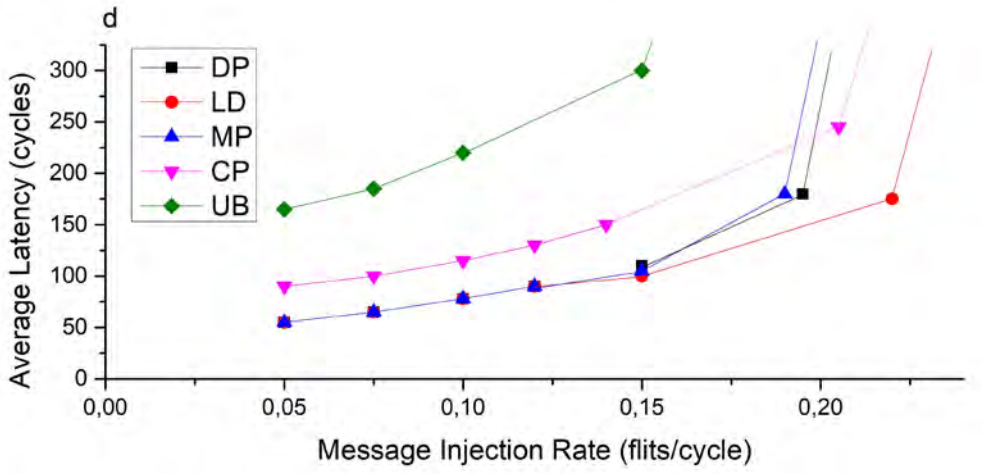
Το πρώτο σετ προσομοιώσεων πραγματοποιήθηκε για τυχαίο προφίλ κυκλοφορίας. Σε αυτές τις προσομοιώσεις, οι PE παράγουν μηνύματα δεδομένων και εισάγουν τους στο δίκτυο χρησιμοποιώντας τα χρονικά διαστήματα που λαμβάνονται με την εκθετική κατανομή. Έχουν ληφθεί υπόψη δύο μεγέθη NoC 8×8 και 16×16. Στο προφίλ multicast, κάθε PE στέλνει ένα μήνυμα σε ένα σύνολο προορισμών. Μια ομοιόμορφη κατανομή χρησιμοποιήθηκε για την κατασκευή του συνόλου προορισμών κάθε μηνύματος πολλαπλής διανομής [12]. Ο αριθμός προορισμών ορίστηκε σε 10 και 25.

Στην εικόνα 2.23 φαίνεται η μέση καθυστέρηση επικοινωνίας ως συνάρτηση του μέσου flit injection. Όπως δείχνουν τα αποτελέσματα, ο αλγόριθμος δρομολόγησης LD multicast οδηγεί στη χαμηλότερη καθυστέρηση μεταξύ των τριών αλγορίθμων δρομολόγησης multicast ακόμη και σε φορτία υψηλής κυκλοφορίας ή με μεγάλο αριθμό προορισμών (25 προορισμούς).



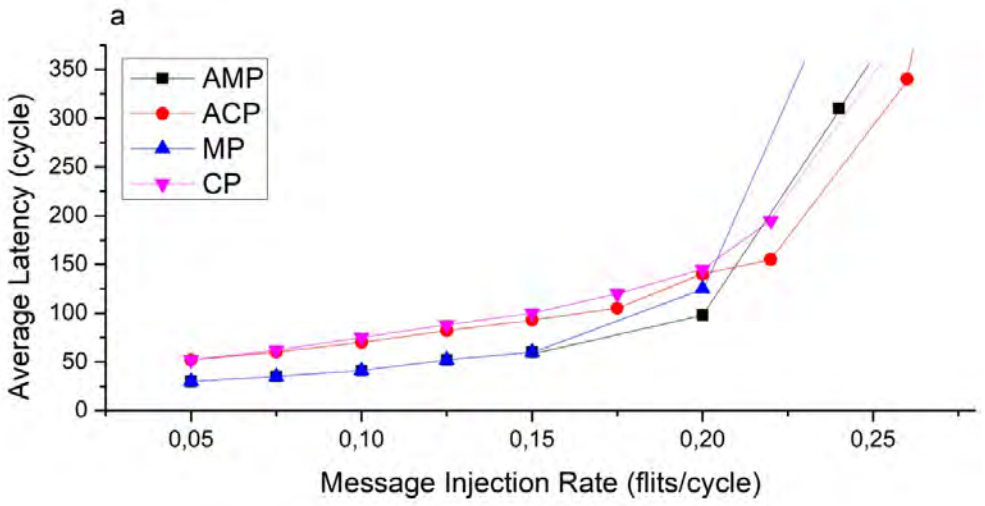


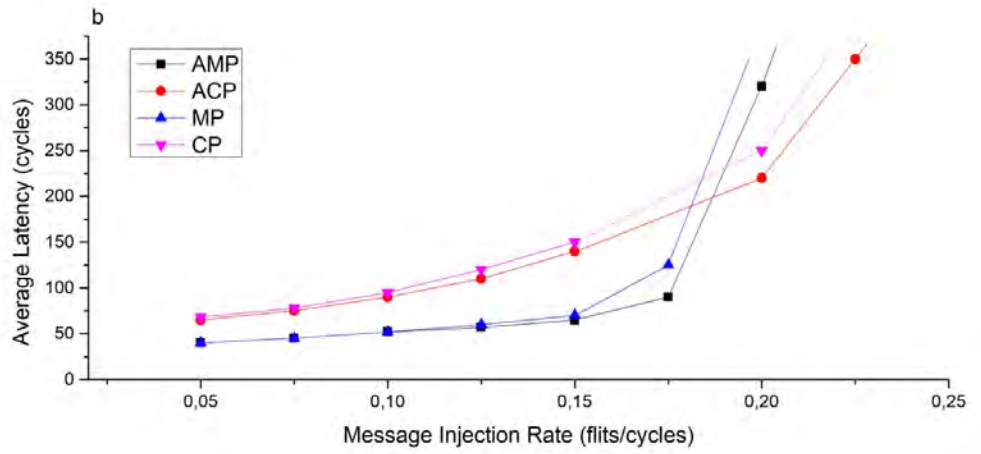


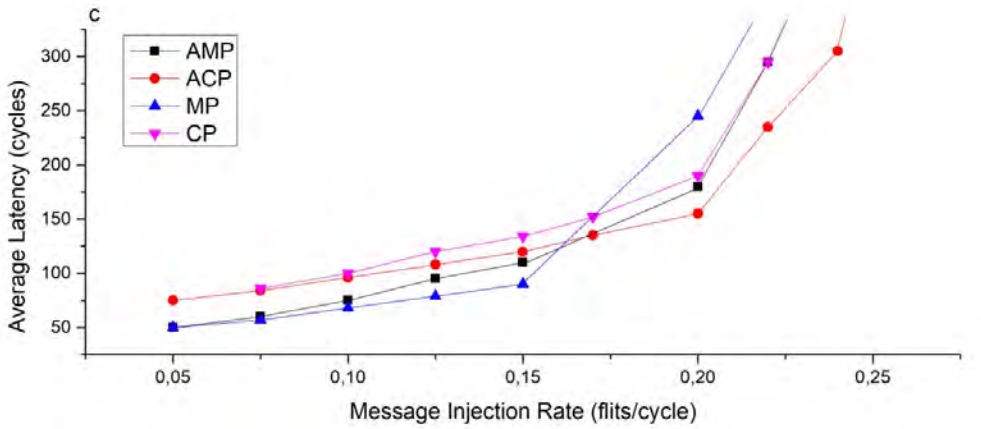


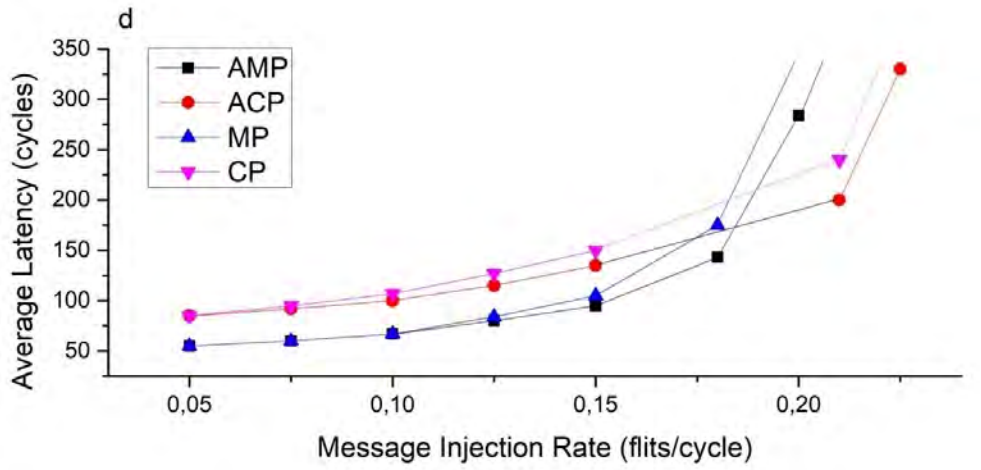
Εικόνα 2.23 Αξιολόγηση επιδόσεων υπό διαφορετικά φορτία σε 8×8 2D-mesh με α) 10 προορισμούς, β) 25 προορισμούς και σε 16×16 2D-mesh γ) 10 προορισμών, δ) multicast traffic.

Η εικόνα 2.24 δείχνει το κέρδος απόδοσης της χρήσης του HAMUM με διαφορετικό αριθμό προορισμών και μεγεθών flits. Όπως παρατηρούμε από τα αποτελέσματα, ο προσαρμοστικός μηχανισμός εφαρμόστηκε στα συστήματα MP και CP ακόμη και σε φορτία μεγάλης κυκλοφορίας ή με μεγάλο αριθμό προορισμών οδηγεί σε μικρότερη καθυστέρηση.





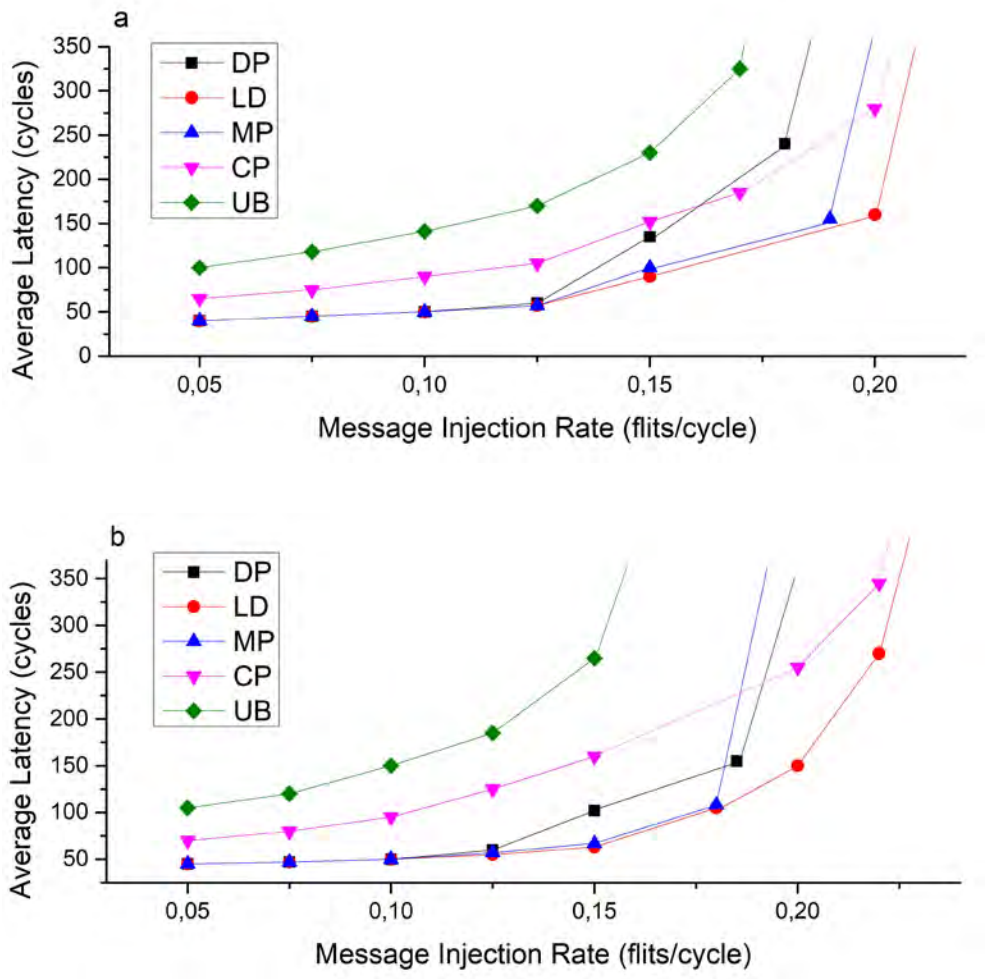




Εικόνα 2.24 Αξιολόγηση απόδοσης του HAMUM υπό διαφορετικά φορτία σε 8×8 2D mesh με α) 10 προορισμούς, β) 25 προορισμών και σε 16×16 mesh γ) 10 προορισμών, δ) 25 προορισμών με multicast traffic.

2.3.5.2 Unicast και Multicast (Mixed) Traffic Profiles

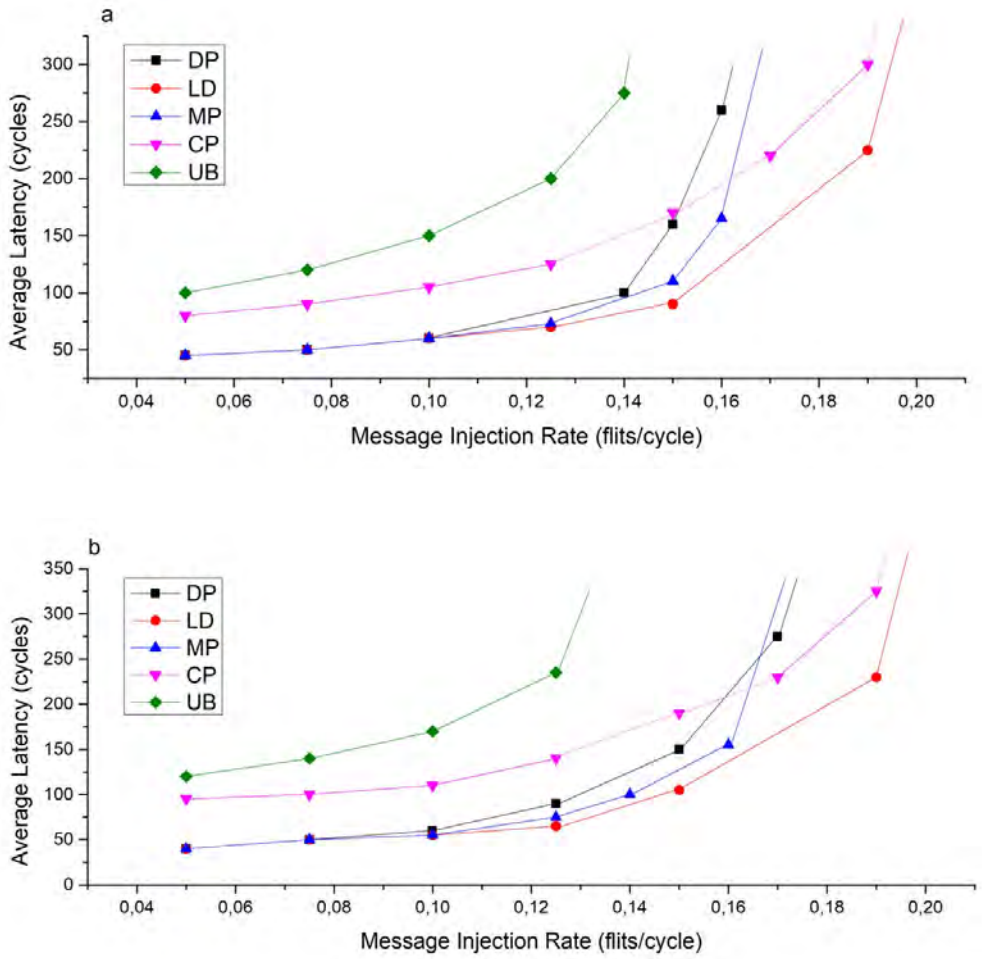
Σε αυτές τις προσομοιώσεις, χρησιμοποιούμε ένα μείγμα unicast και multicast διανομής, όπου το 80% των μηνυμάτων που εγχύθηκαν είναι μηνύματα unicast και το υπόλοιπο 20% είναι μηνύματα πολλαπλής διανομής. Αυτό το μοτίβο μπορεί να αντιπροσωπεύει την κίνηση σε ένα πολυεπεξεργαστή κατανεμημένης κοινής μνήμης, όπου οι ενημερώσεις και η invalidation παράγουν μηνύματα πολλαπλής διανομής και η προσωρινή μνήμη εξυπηρετείται από τα μηνύματα unicast [15] [19]. Για αυτό το σύνολο, οι παράμετροι προσομοίωσης είναι ίδιες με τις προηγούμενες προσομοιώσεις από την άποψη του αριθμού των προορισμών. Τα μηνύματα unicast δρομολογούνται επίσης με το μοντέλο περιστροφής του Odd-Even. Uniform και hotspot [8] είναι τα δύο διαφορετικά χαρακτηριστικά κυκλοφορίας που εξετάστηκαν για την παραγωγή κυκλοφορίας unicast. Στο ομοιόμορφο προφίλ κίνησης, κάθε PE στέλνει ένα μήνυμα σε οποιοδήποτε άλλο PE με ίσες πιθανότητες. Επομένως, οι προορισμοί καθορίζονται τυχαία με ομοιόμορφη κατανομή. Κάτω από το πρότυπο κυκλοφορίας hotspot, ένας ή περισσότεροι κόμβοι επιλέγονται ως hotspots που λαμβάνουν ένα επιπλέον τμήμα της κυκλοφορίας εκτός από την κανονική ομοιόμορφη κυκλοφορία. Στην εικόνα 2.25, εμφανίζονται οι μέσες καθυστερήσεις επικοινωνίας σε σχέση με το ρυθμό έγχυσης μηνυμάτων για διαφορετικούς αλγόριθμους κάτω από το ομοιόμορφο μοντέλο κυκλοφορίας για το προφίλ κίνησης Unicast. Όπως βλέπουμε από τα σοιχεία αυτά, LD ξεπερνά τους άλλους αλγόριθμους.



Εικόνα 2.25. Αξιολόγηση των επιδόσεων σε διαφορετικά φορτία σε 8×8 2D-mesh με α) 10 προορισμούς, β) 25 προορισμοί με μικτή κίνηση (20% multicast και 80% unicast) ενώ η κυκλοφορία unicast βασίζεται στο uniform traffic model.

Κάτω από το μοντέλο κυκλοφορίας hotspot, ένα μήνυμα που δημιουργείται κατευθύνεται σε κάθε κόμβο hotspot με επιπλέον πιθανότητα h τοις εκατό. Στις προσομοιώσεις μας, τοποθετήσαμε έναν μοναδικό κόμβο hotspot (κόμβος (4, 4)). Η εικόνα 2.26 δείχνει τις μέσες λανθάνουσες περιόδους των αλγορίθμων για τις τοπολογίες των 8×8 2D πλέγματος όταν h = 10%. Όπως δείχνουν τα διαγράμματα, το LD ξεπερνά σημαντικά τους άλλους

αλγόριθμους για διαφορετικούς αριθμούς προορισμών κάτω από διαφορετικούς ρυθμούς έγχυσης μηνυμάτων.

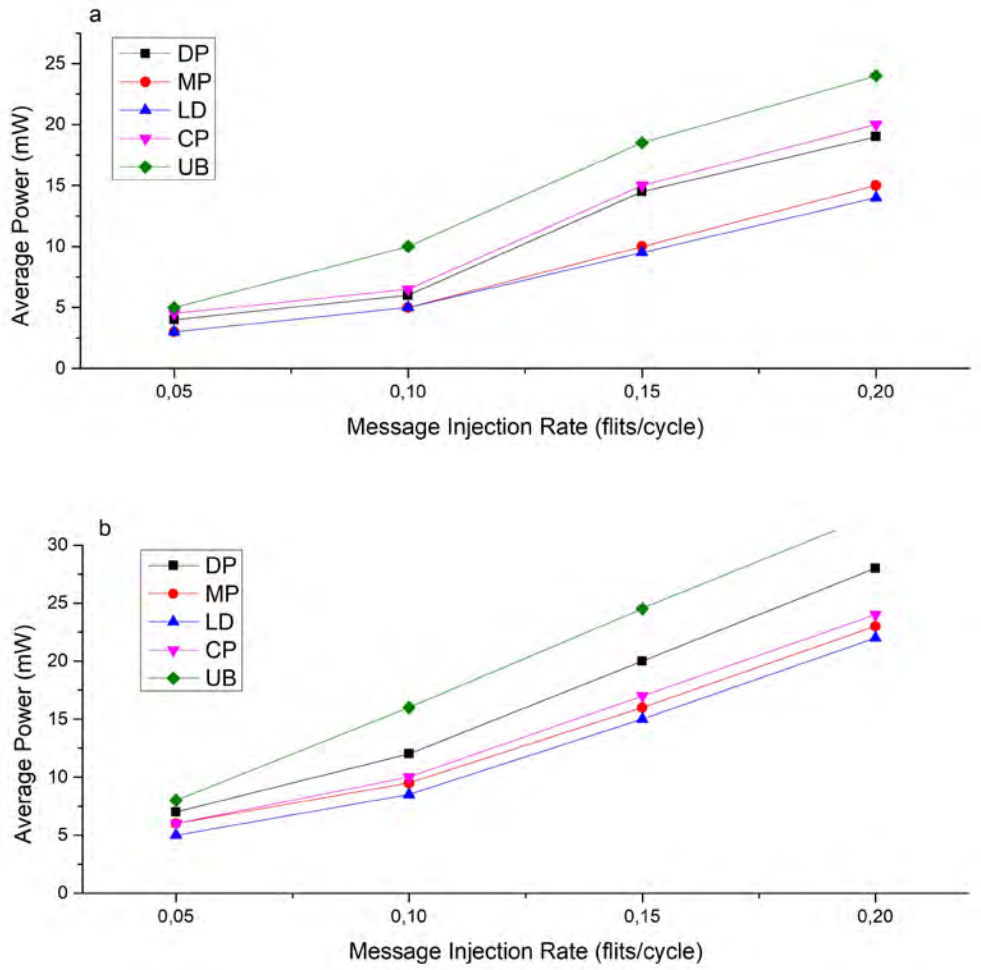


Εικόνα 2.26 Απόδοση αλγορίθμων κάτω από διαφορετικά φορτία σε 8×8 2D-mesh με α) 10 προορισμούς, β) 25 προορισμούς με μικτή κίνηση (20% multicast και 80% unicast). Η κυκλοφορία Unicast βασίζεται στο μοντέλο κυκλοφορίας hotspot με έναν μοναδικό κόμβο hotspot (4, 4). Το ποσοστό hotspot είναι 10 τοις εκατό.

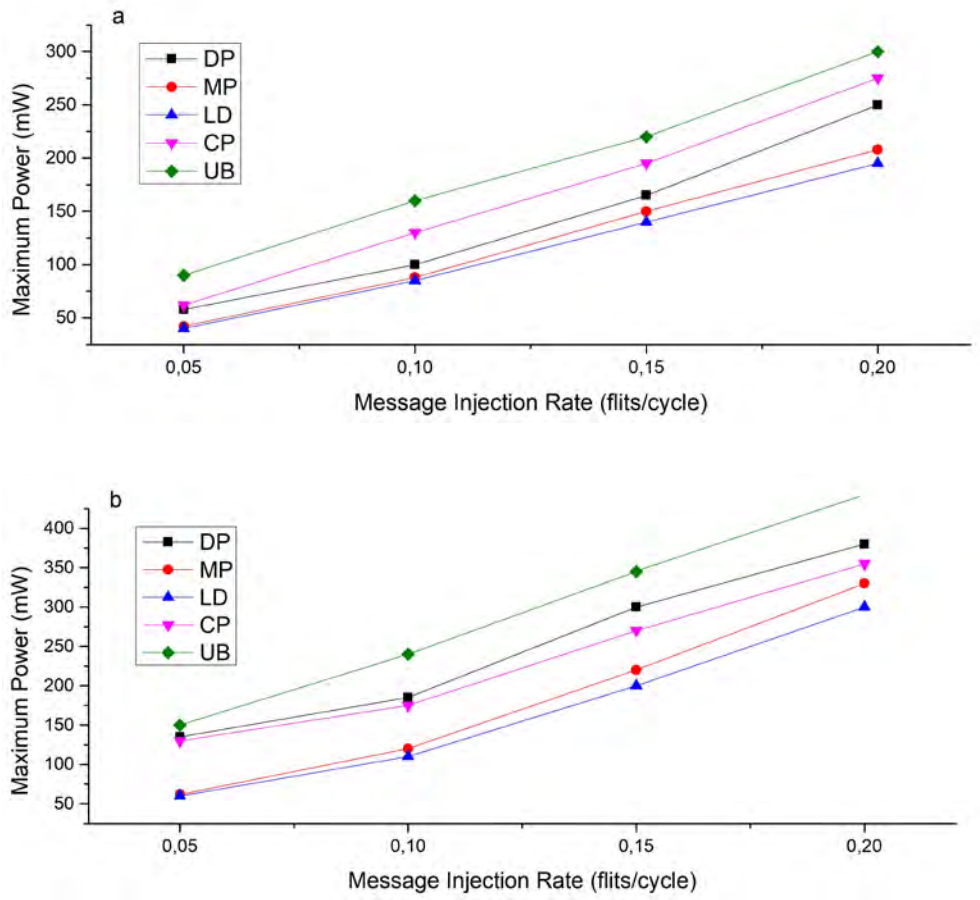
2.3.5.3 Power consumption

Η κατανάλωση ισχύος των DP, MP, CP, UB και LD αλγορίθμων δρομολόγησης υπολογίστηκαν και συγκρίθηκαν με το μοντέλο κυκλοφορίας

πολλαπλών μεταδόσεων. Τα αποτελέσματα για τη μέση και τη μέγιστη ισχύ κάτω από αυτή την κυκλοφορία παρουσιάζονται στις εικόνες 2.27 και 2.28, αντίστοιχα.



Εικόνα 2.27. Μέση καταναλισκόμενη ισχύς των DP, MP και CP αλγορίθμων σε 16×16 2D-mesh με α) 10 προορισμούς και β) 25 προορισμών υπό κυκλοφορία πολλαπλών μεταδόσεων.



Εικόνα 2.28. Μέγιστη απορρόφηση ισχύος των προτεινόμενων, των DP, MP και CP αλγορίθμων σε 16×16 2D-mesh με α) 10 προορισμούς και β) 25 προορισμών υπό κυκλοφορία πολλαπλών μεταδόσεων.

Όπως μπορούμε να δούμε στα αποτελέσματα που παρουσιάζονται στον Πίνακα 2.5 για 10 προορισμούς, η μέση κατανάλωση ισχύος του δικτύου με τον αλγόριθμο LD είναι 25%, 3,5%, 33% και 63% λιγότερα από τα DP, MP, CP και UB στο μοντέλο κυκλοφορίας πολλαπλών μεταδόσεων, αντίστοιχα. Επίσης, τα αποτελέσματα του Πίνακα 2.6 για 10 προορισμούς υποδεικνύουν ότι η μέγιστη ισχύς του αλγορίθμου LD είναι 27%, 8%, 44% και 70% λιγότερες από εκείνες των αλγορίθμων DP, MP, CP και UB αντίστοιχα, στο μοντέλο κυκλοφορίας πολλαπλών μεταδόσεων. Παρόμοια εξοικονόμηση ενέργειας επιτυγχάνεται για 25 προορισμούς. Η μείωση της ισχύος για τον

αλγόριθμο LD επιτυγχάνεται με την ομαλή κατανομή της κατανάλωσης ενέργειας μέσω του δικτύου, χρησιμοποιώντας το προσαρμοστικό σύστημα δρομολόγησης, το οποίο μειώνει τον αριθμό των ενεργών σημείων πρόσβασης και επομένως μειώνει τόσο τη μέση ισχύ όσο και τη μέγιστη ισχύ. Ο πίνακας 2.7 αποκαλύπτει ότι η μέση απορρόφηση ισχύος του δικτύου με τον αλγόριθμο ACP είναι 5% μικρότερη από αυτή του αλγόριθμου CP και η μέση κατανάλωση ισχύος του AMP είναι 3,5% μικρότερη από αυτή του αλγόριθμου MP. Τα αποτελέσματα του πίνακα 2.8 υποδεικνύουν ότι η μέγιστη ισχύς των αλγορίθμων ACP και AMP είναι 15% και 11% μικρότερη από εκείνη των αλγορίθμων CP και MP, αντίστοιχα κάτω από το μοντέλο κυκλοφορίας πολλαπλών μεταδόσεων. Μπορούμε να παρατηρήσουμε ότι η μέση ισχύς και η μέγιστη ισχύς των προτεινόμενων προσαρμοστικών μοντέλων είναι χαμηλότερα.

Average Power Dissipation	DP	MP	CP	UB
With 10 Destinations	-25%	-3.5%	-33%	-63%
With 25 Destinations	-32%	-8.5%	-13%	-51%

Πίνακας 2.5. Συγκριτικός πίνακας μέσης απορρόφησης ισχύος του LD σε σχέση με άλλους αλγόριθμους σε 16×16 2D-mesh.

Average Power Dissipation	DP	MP	CP	UB
With 10 Destinations	-27%	-8%	-44%	-70%
With 25 Destinations	-43%	-12%	-33%	-64%

Πίνακας 2.6. Συγκριτικός πινακας μέγιστης απορρόφησης ισχύος του LD σε σχέση με άλλους αλγορίθμους σε 16×16 2D-mesh.

Average Power Dissipation	AMP/MP	ACP/CP
With 25 Destinations	3.5%	5%

Πίνακας 2.7. Συγκριτικός πίνακας μέσης απορρόφησης ισχύος των προσαρμοστικών σχημάτων χρησιμοποιώντας μοντέλο HAMUM σε σχέση με τα συμβατικά σχήματα.

Average Power Dissipation	AMP/MP	ACP/CP
With 25 Destinations	11%	15%

Πίνακας 2.8. Συγκριτικός πίνακας μέγιστης κατανάλωσης ισχύος των προσαρμοστικών σχημάτων χρησιμοποιώντας το μοντέλο HAMUM σε σχέση με τα συμβατικά σχήματα.

2.4 Συμπεράσματα

Τα πρωτόκολλα δρομολόγησης μπορούν να έχουν μεγάλο αντίκτυπο στην απόδοση και την κατανάλωση ενέργειας στα δίκτυα onchip. Τα πρωτόκολλα δρομολόγησης σε NoCs μπορούν να είναι είτε unicast (one-to-one) είτε multicast (one-to-many). Το πρώτο πρωτόκολλο δρομολόγησης που παρουσιάστηκε ήταν ένας προσαρμοστικός αλγόριθμος δρομολόγησης με γνώμονα την κυκλοφοριακή συμφόρηση για δικτυωτούς διαύλους NoCs, ο οποίος δεν υποστηρίζει κυκλοφορία πολλαπλής διανομής, ενώ τα άλλα δύο πρωτόκολλα παρουσιάζουν προσαρμοστικά μοντέλα δρομολόγησης που υποστηρίζουν τόσο την κίνηση unicast όσο και την multicast.

Ο πρώτος αλγόριθμος δρομολόγησης είναι ένας προσαρμοστικός αλγόριθμος δρομολόγησης, ο οποίος ονομάζεται EDXY, ο οποίος βελτιώνει τον αλγόριθμο δρομολόγησης DyXY. Σε αυτήν την τεχνική, προστέθηκαν δύο σύρματα συμφόρησης στην αρχιτεκτονική του δρομολογητή για να σηματοδοτήσουν τη συμφόρηση της γραμμής ή της στήλης. Αυτό επέτρεψε την αποφυγή της συμφόρησης διαδρομής, και έτσι να μειωθεί η καθυστέρηση του αλγορίθμου. Επιπλέον, παρουσιάστηκαν δύο προσαρμοστικά πρωτόκολλα δρομολόγησης (LD και HAMUM) για δικτυακά δίκτυα που βασίζονται σε αρχιτεκτονική mesh. Το LD χρησιμοποιεί διαχωρισμό σε υποδίκτυα και τον προσαρμοστικό αλγόριθμο του μοντέλου Odd-Even για τη δρομολόγηση τόσο των μηνυμάτων multicast όσο και unicast μέσω του δικτύου. Επιπλέον, ο προσαρμοστικός αλγόριθμος δρομολόγησης χρησιμοποίησε την κατάσταση συμφόρησης των θυρών εισόδου για τη

δρομολόγηση των μηνυμάτων μέσω non faulty διαδρομών. Ωστόσο, η εκμετάλλευση των αλγορίθμων δρομολόγησης unicast για κυκλοφορία πολλαπλών μεταδόσεων μπορεί να αυξήσει την καθυστέρηση. Το HAMUM παρουσιάστηκε ως προσαρμοστική μέθοδος δρομολόγησης τόσο για την κυκλοφορία unicast όσο και για την mulyicast, η οποία μεγιστοποιεί τον βαθμό προσαρμοστικότητας των λειτουργιών δρομολόγησης που βασίζονται στη διαδρομή Hamiltonian.

Βιβλιογραφία

1. W. O. Cesario, D. Lyonnard, G. Nicolescu, Y. Paviot, S. Yoo, A. A. Jerraya, L. Gauthier, M. Diaz-Nava, “Multiprocessor SoC platforms: a component-based design approach,” in proc. of IEEE Design and Test of Computers, pp. 52–63, 2002, France.

2. E. Rijpkema, K.G.W. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage, and E. Waterlander, “Trade offs in the design of a router with both guaranteed and besteffort services for networks on chip,” in proc. of IEEE Design Automation and Test Conference in Europe (DATE), pp. 350–355, Germany 2003.

3. A. Jantsch and H. Tenhunen, “Networks on Chip,” Kluwer Academic Publishers, 2003.

4. W. J. Dally and B. Towles, “Principles and Practices of Interconnection Networks,” Morgan Kaufmann, 2004.

5. W. J. Dally and B. Towles, “Route Packets, Not Wires: On-Chip Interconnection Networks,” In Proc. of Design Automation Conference (DAC), pp. 684-689, June 2001.

6. J.C. Hu, R. Marculescu, “DyAD–smart routing for networks-on-chip,” in proc. of the Design Automation Conference (DAC), pp. 260–263, USA 2004.

7. J. Kim, D. Park, T. Theocharides, N. Vijaykkrishnan, and C.R. Das, “A low latency router supporting adaptivity for on-chip interconnects,” in proc. of International Symposium on Computer Architecture, pp. 150–161, USA 2005.

8. G. M. Chiu, “The odd-even turn model for adaptive routing,” IEEE Trans. on Parallel and Distributed Systems, vol. 11, pp. 729 – 738, 2000.

9. J. Henkel, W. Wolf, S. Chakradhar, “On-chip networks: A scalable, communication-centric embedded system design paradigm,” in proc. of 17th International Conference on VLSI Design (VLSID), pp.845, India, 2004.

10. S. Vangal et al., “An 80-tile 1.28TFlops Network-on-Chip in 65nm CMOS,” In Proceedings of ISSCC’07, pp. 98-100, 2007.

11. P. Gratz, B. Grot, S.W. Keckler, “Regional congestion awareness of load balance in network-on-chips,” in proc. International Symposium on High Performance Computer Architecture, pp. 203–214, February 2008.

12. M. Li, Q.-A. Zeng, and W.-B. Jone, “DyXY - a proximity congestion-aware deadlock-free dynamic routing method for network on chip,” in Proc. of Design Automation Conference, pp. 849–852, 2006.

13. J.C. Hu, R. Marculescu, “DyAD–smart routing for networks-on-chip,” in proc. of the Design Automation Conference (DAC), pp. 260–263, USA 2004.

14. P. McKinely, H. Xu, A. H. Esfahanian, and L. Ni, “Unicast-based multicast communication in wormhole-routed networks,” IEEE Trans. of Parallel and Distributed Systems, Vol. 5, pp. 1252-1265, 1994.

15. M.P. Malumbres, J. Duato, and J. Torrellas, “An Efficient Implementation of Tree-Based Multicast Routing for Distributed Shared-Memory Multiprocessors,” in proc. of SPDP, pp. 186-190, USA, 1996.

16. A. Al-Dubai, I. Romdhani, “A Performance Study of Path Based Multicast Communication Algorithms,” in proc. int. conf. PARELEC, Bialystok, Poland, pp. 245-250, 2006.

17. X. Lin, and L. M. Ni, “Multicast Communication in Multicomputer Networks,” IEEE Transactions on Parallel and Distributed Systems, 4, pp. 1105-1117, 1993.

18. N. E. Jerger, L. S. Peh, and M. H. Lipasti, “Virtual Circuit Tree Multicasting: A Case for On-Chip Hardware Multicast Support,” in proc. of International Symposium Computer Architecture (ISCA), pp. 229-240, China 2008.

19. R. V. Boppana, , S. Chalasani, C.S. Raghavendra, “Resource deadlock and performance of wormhole multicast routing algorithms,” IEEE Transactions on Parallel and Distributed Systems, 9, pp. 535-549, 1998,

20. A. Al-Dubai, M. Ould-Khaoua, and L. M. Mackenzie “An Efficient Path-Based Multicast Algorithm for Mesh Networks,” in proc. of IPDPS, pp. 1—8, 2003.

21. F. Harary, ‘Graph Theory’ (Readings, Massachusetts: Addson-Wesley, 1972)

22. L.M. Ni, P.K. McKinley, “A survey of wormhole routing techniques in direct networks,” IEEE Computer, Vol. 26, No. 2, pp. 62–76, 1993.

23. A. Kahng, B. Li, L.-S. Peh, and K. Samadi, “Orion 2.0: A fast and accurate noc power and area model for early-stage design space exploration,” in proc. of DATE, pp. 423-428, 2009.

3 ο ΚΕΦΑΛΑΙΟ

FAULT-TOLERANT ROUTING ALGORITHMS ON NETWORKS ON-CHIP

3.1 Εισαγωγή

Τις τελευταίες δύο δεκαετίες έχουν προταθεί πολλές προσεγγίσεις για να καταστούν τα δίκτυα διασύνδεσης ανεκτικά στα σφάλματα (fault tolerant). Αυτές οι προσεγγίσεις μπορούν να χωριστούν σε δύο κατηγορίες: στις αρχιτεκτονικές που είναι ανεκτικές στα σφάλματα και στους αλγορίθμους που είναι ανεκτικές σε σφάλματα. Η πρώτη κατηγορία αντιμετωπίζει το σχεδιασμό ανθεκτικότητας σε σφάλματα των εξαρτημάτων on-chip ώστε να λειτουργούν κανονικά με την παρουσία ελαττωμάτων.

Οι διαδρομές ανεκτικών σφαλμάτων λαμβάνουν local και global αποφάσεις σχετικά με τη διαδρομή των πακέτων, με βάση τη θέση των ελαττωματικών συνδέσμων και κόμβων. Σε αυτή την περίπτωση, η εύρεση νέων διαδρομών για τα πακέτα μπορεί να γίνει είτε centralized είτε distributed [1, 2]. Οι κατανεμημένοι αλγόριθμοι κατανέμουν το γενικό υπολογιστικό κόστος για την εύρεση νέων διαδρομών μεταξύ όλων των κόμβων του δικτύου, ενώ σε έναν συγκεντρωτικό αλγόριθμο, ένας ή περισσότεροι κόμβοι είναι υπεύθυνοι

για την εκ νέου δρομολόγηση των πακέτων. Έτσι, η αρχιτεκτονική των δρομολογητών που χρειάζονται έναν κατανεμημένο αλγόριθμο θα μπορούσε να είναι απλούστερη από τους δρομολογητές που χρησιμοποιούν έναν και μοναδικό κεντρικό αλγόριθμο. Ωστόσο, η υπολογιστική ισχύς που πρέπει να εκτελεστεί σε έναν κεντρικό αλγόριθμο είναι πολύ μεγαλύτερη από τους υπολογισμούς που γίνονται σε έναν κατανεμημένο αλγόριθμο που εκτελείτε ξεχωριστά σε κάθε κόμβο. Επιπλέον, οι κεντρικοί αλγόριθμοι χρησιμοποιούν πληροφορίες global fault, αλλά οι κατανεμημένοι αλγόριθμοι γνωρίζουν την εμφάνιση σφάλματος μόνο βάσει τοπικών πληροφοριών από τους γειτονικούς κόμβους. Αυτό οφείλεται κυρίως στο γεγονός ότι η αποστολή των γενικών πληροφοριών σφαλμάτων σε όλους τους κόμβους καταναλώνει ένα μεγάλο μέρος των πόρων του συστήματος. Ως εκ τούτου, η απόφαση επιλογής διαδρομής σε συγκεντρωτικούς αλγόριθμους μπορεί να οδηγήσει σε αποτελέσματα καλύτερης ποιότητας.

Σε αυτό το κεφάλαιο, παρουσιάζουμε τους βασικούς μηχανισμούς δρομολόγησης που είναι ανεκτοί σε σφάλματα (FaulToleReR) για να αντιμετωπίσουμε τα σφάλματα στα NoCs. Αστοχίες στο σύστημά μας προκύπτουν υπό ορισμένες συνθήκες. Κυρίως οι διακυμάνσεις της διαδικασίας σε συνδυασμό με τις απαιτητικές λειτουργίες του NoC (όπως η υψηλή θερμοκρασία [3]) επηρεάζουν το σύστημα όσο ο αλγόριθμος συνεχίζει τη λειτουργία του [4]. Το ελαττωματικό μέρος θα συνεχίσει την κανονική του λειτουργία όταν οι συνθήκες (θερμοκρασία, για παράδειγμα) επιστρέψουν στο φυσιολογικό. Η θερμοκρασία αυξάνει την καθυστέρηση διάδοσης αν υπερβαίνει ένα όριο. Το θερμό στοιχείο μπορεί να μην ολοκληρώσει τον υπολογισμό και την επικοινωνία μέσα στον κύκλο ενός μόνο ρολογιού, οδηγώντας σε δημιουργία σφαλμάτων στις εξόδους των NoC [5, 6].

Αυτός ο μηχανισμός δρομολόγησης έχει ως στόχο να διαχειρίζεται τις διαδρομές των πακέτων σε ένα NoC, στα οποία ορισμένα τμήματα γίνονται

προσωρινά μη λειτουργικά λόγω ορισμένων συνθηκών, ιδιαίτερα της υψηλής θερμοκρασίας. Η θερμοκρασία των συνδέσμων και άλλων εξαρτημάτων του NoC σχετίζεται άμεσα με τον φόρτο κυκλοφορίας σε αυτά. Κατά συνέπεια, τα intermittent σφάλματα που προκαλούνται από την υψηλή θερμοκρασία θα εξαλείφουν όταν δεν υπάρχει φόρτος σε έναν ελαττωματικό σύνδεσμο για ένα χρονικό διάστημα. Το πιο απλό μοντέλο που μπορούμε να χρησιμοποιήσουμε για τη θερμοκρασία κάθε εξαρτήματος είναι να υποθέσουμε ότι αυτή είναι ανάλογη της κατανάλωσης ενέργειας του εξαρτήματος. Σε αυτή την περίπτωση δεχόμαστε ότι η θερμοκρασία κοντινών εξαρτημάτων δεν επηρεάζει σε ανησυχητικό βαθμό τη θερμοκρασία του εξαρτήματος που εξετάζουμε.

Η τοπολογία του NoC σε αυτή την περίπτωση αλλάζει δυναμικά κατά τη διάρκεια της ζωής του συστήματος λόγω των συνδέσεων που επηρεάζονται από intermittent βλάβες. Το FaulToleReR στοχεύει σε μια τέτοια κατάσταση και προσπαθεί να αλλάξει δυναμικά τον αλγόριθμο δρομολόγησης κατά το χρόνο εκτέλεσης. Δρομολογεί τα πακέτα με τέτοιο τρόπο ώστε όλοι τους να προωθούνται από την πιθανότερη συντομότερη διαδρομή μεταξύ των κόμβων προέλευσης και προορισμού τους, η κυκλοφορία είναι ισορροπημένη σε όλο το τσιπ, ο περιορισμός του εύρους ζώνης κάθε συνδέσμου ικανοποιείται και η αποφυγή του αδιεξόδου είναι εγγυημένη.

Όπως οι περισσότερες εφαρμογές σε NoC, έτσι και στη συγκεκριμένη περίπτωση, ο μηχανισμός δρομολόγησης είναι βασισμένος σε πίνακα για να μπορούμε να εκμεταλλευτούμε στο έπακρο πληροφορίες σχετικά με τα χαρακτηριστικά της δρομολόγησης, τυχόν ελαττωματικούς συνδέσμους στο NoC, εναλλακτικές διαδρομές δεδομένων κ.α. Αναλύσεις [7] πιστοποιούν ότι αλγόριθμοι δρομολόγησης βασισμένοι σε πίνακα είναι καταλληλότεροι για συστήματα NoC λόγω του χαμηλότερου κόστους, της ευκολίας υλοποίησης και της αποδοτικότητας ισχύος.

Οι περισσότεροι αλγόριθμοι δρομολόγησης fault-tolerant χρησιμοποιούν εικονικά κανάλια (VC) για να εφαρμόσουν τη μέθοδο του καναλιού διαφυγής για να αποφευχθεί το αδιέξοδο [8], ενώ κάποιες άλλες μέθοδοι επιτυγχάνουν την αποφυγή αδιεξόδου τοποθετώντας ορισμένους περιορισμούς στα μονοπάτια που μπορούν να ληφθούν από τα πακέτα. Η τελευταία περίπτωση οδηγεί σε συστήματα με λιγότερη κατανάλωση ενέργειας, πολυπλοκότητα και κόστος, σε σύγκριση με τις μεθόδους αποφυγής αδιεξόδων που βασίζονται στα VC.

Οι intermittent βλάβες εμφανίζονται συχνά και διαρκούν αρκετούς κύκλους. Ως εκ τούτου, ένας αλγόριθμος δρομολόγησης πρέπει να προσαρμόζεται σε τέτοια συχνά εμφανιζόμενα σφάλματα. Οι ντετερμινιστικοί fault-tolerant αλγόριθμοι δρομολόγησης θα πρέπει να μπορούν να προσαρμοστούν σύμφωνα με αλλαγές τοπολογίας για να αποφευχθούν τέτοιου είδους σφάλματα. Ωστόσο, οι υπάρχουσες μέθοδοι προσαρμοστικής δρομολόγησης χρησιμοποιούν τοπικές πληροφορίες για να επιλέξουν μια νέα διαδρομή. Σε αυτήν την περίπτωση, η περιορισμένη τοπική πληροφόρηση μπορεί να οδηγήσει σε λανθασμένες αποφάσεις.

Επωφελούμενοι από τις global πληροφορίες σχετικά με τα fault patterns και τη ζήτηση επικοινωνίας κάθε κόμβου με χαμηλή overhead είναι το βασικό πλεονέκτημα για τις λίγες εργασίες που χειρίζονται intermittent σφάλματα [3].

Παρόλο που το FaulToleReR στοχεύει σε intermittent σφάλματα, μπορεί να χρησιμοποιηθεί για τη δρομολόγηση πακέτων σε οποιοδήποτε δίκτυο με δυναμική τοπολογία (όπως ασύρματα δίκτυα), καθώς και σε δίκτυα παράκαμψης με υπερμεγέθη IPs (OIPs) [9].

3.2 Ανάλυση του προβλήματος

Όπως αναφέρθηκε προηγουμένως, η μη χρησιμοποίηση συγκεκριμένων συνδέσμων για ορισμένο χρονικό διάστημα προκαλεί δυναμικές αλλαγές στην τοπολογία NoC. Η κίνηση υποτίθεται επίσης ότι είναι δυναμική, οπότε ο αρχικός κόμβος χρησιμοποιεί τις συλλεγμένες πληροφορίες σχετικά με την κίνηση (σε σχέση με τον κόμβο προορισμού και το βάρος της κάθε ροής επικοινωνίας) σε κάθε χρονικό σημείο για να δημιουργήσει ένα γράφημα που θα χρησιμοποιηθεί κατά τον υπολογισμό της διαδρομής διαδικασία. Οι πληροφορίες για τους μη διαθέσιμους συνδέσμους συλλέγονται επίσης από τον αρχικό κόμβο μέσω του δικτύου ελέγχου. Στόχος μας είναι να αλλάξουμε τον αλγόριθμο δρομολόγησης προκειμένου να δρομολογήσουμε αποτελεσματικά τα πακέτα με βάση τη νέα centralized τοπολογία.

Θεωρείται ότι οι κόμβοι που είναι μη επιτρεπτοί δεν επηρεάζουν τη συνδεσιμότητα δικτύου, δηλαδή υπάρχει τουλάχιστον μία διαδρομή μεταξύ οποιωνδήποτε δύο κόμβων του NoC που πρέπει να επικοινωνούν μεταξύ τους. Διαφορετικά, είναι αδύνατο να βρεθεί διαδρομή για ορισμένα πακέτα τα οποία με τη σειρά τους οδηγούν σε αποτυχία του συστήματος.

Μοντελοποιούμε όλα τα σφάλματα ως σφάλματα σύνδεσης, είτε αυτά συμβούν στον δρομολογητή είτε στην ίδια την σύνδεση. Το σύνολο των ελαττωματικών συνδέσεων περιλαμβάνει όλους τους συνδέσμους του NoC που δεν είναι διαθέσιμοι αυτήν τη στιγμή. Για να διατυπώσουμε το πρόβλημα, πρέπει πρώτα να δώσουμε τους ακόλουθους ορισμούς:

Ορισμός 1 Communication Task Graph: Μια εργασία και οι αντίστοιχες σχέσεις αντιπροσωπεύονται ως γράφημα εργασιών επικοινωνίας, CTG που είναι ένα κατευθυνόμενο γράφημα G (V, E) όπου κάθε $v_i \in V$ αντιπροσωπεύει μια εργασία και ένα κατευθυνόμενο γράφο, το $e_{i,j} \in E$ χαρακτηρίζει μια ροή επικοινωνίας από το v_i στο v_j. Ο όγκος επικοινωνίας

(bit ανά δευτερόλεπτο) που αντιστοιχεί σε κάθε άκρο $e_{i,\ j}$ αντιπροσωπεύεται από $t_{(ei,\ j)}$.

Ορισμός 2 Περιορισμός Bandwidth Link: Ο περιορισμός του εύρους ζώνης κάθε σύνδεσης l_k του NoC θα πρέπει να ικανοποιηθεί ως εξής:

$$\forall l_k, BW\left(l_k\right) \geq \sum_{\forall e_{i,j} \in E} x^k\left(i, j\right)$$

Όπου το $BW_{(lk)}$ χαρακτηρίζει το εύρος ζώνης της σύνδεσης l_k και το $X^k(i, j)$ υπολογίζεται ως εξής:

$$\mathrm{X}^k\left(i, j\right) = \begin{cases} t(e_{i,j}) & \alpha \nu\, l_k \in path\left(e_{i,j}\right) \\ 0 & \sigma\varepsilon\ o\pi o\iota\alpha\delta\acute{\eta}\pi o\tau\varepsilon\ \acute{\alpha}\lambda\lambda\eta\ \pi\varepsilon\rho\acute{\iota}\pi\tau\omega\sigma\eta \end{cases}$$

Η παράμετρος $path(e_{i,j})$ είναι το σύνολο δεσμών στις οποίες χαρτογραφείται η ακμή CTG $e_{i,\ j}$. Αυτό μπορεί να δηλωθεί πιο απλά ως εξής:

$$\forall l_k, BW\left(l_k\right) \geq BR\left(l_k\right) \qquad (1)$$

Όπου BR (l_k) είναι το συνολικό μέγεθος δεδομένων που δρομολογείται στη σύνδεση l_k του NoC.

Τυπικά, το πρόβλημα του δυναμικού υπολογισμού της διαδρομής μπορεί να περιγραφεί ως εξής.

Λαμβάνοντας υπόψη ότι ένα NoC, το οποίο προκύπτει από την αφαίρεση των ελαττωματικών συνδέσμων (faulty links - FL) από τους συνολικούς συνδέσμους του αρχικού NoC και το πρότυπο κυκλοφορίας που περιγράφεται από το CTG, εντοπίζει μια νέα διαδρομή για κάθε ροή (κάθε άκρο $e_{i,j}$ στο CTG) που ελαχιστοποιεί την ακόλουθη έκφραση, όπου η route (i, j) είναι η απόσταση (σε hop) μεταξύ των κόμβων i και j είναι:

$$\sum_{\forall e_{i,j} \in E} t\left(e_{i,j}\right) \times route\left(i, j\right) \quad (2)$$

Η παραπάνω έκφραση μπορεί να θεωρηθεί ως η καθυστέρηση του μέσου μηνύματος του δικτύου, καθώς η καθυστέρηση ενός πακέτου είναι το άθροισμα της καθυστέρησης μηδενικού φορτίου και της καθυστέρησης δέσμευσης. Η καθυστέρηση μηδενικού φορτίου είναι ο χρόνος (σε κύκλους) που θα χρειαστεί για να μεταφερθεί ένα πακέτο από την πηγή στον κόμβο προορισμού όταν όλοι οι πόροι είναι διαθέσιμοι. Ωστόσο, στην πραγματικότητα, ένα πακέτο θα πρέπει να ανταγωνίζεται με άλλα πακέτα για το εύρος ζώνης NoC. Εάν το δίκτυο δεν έχει συμφόρηση, ο αριθμός των hops που διασχίζει ένα πακέτο για να φτάσει στον κόμβο προορισμού είναι κατά προσέγγιση ανάλογος με την καθυστέρηση του πακέτου. Συνεπώς, ελαχιστοποιώντας την παραπάνω εξίσωση θα οδηγήσει στην ελαχιστοποίηση της μέσης καθυστέρησης μηνύματος στο NoC. Η ισχύς που καταναλώνεται για τη μετάδοση πακέτων είναι επίσης ανάλογη με το μήκος διαδρομής πακέτων. Συνεπώς, καθορίζοντας τις διαδρομές των πακέτων με τέτοιο τρόπο ώστε κάθε πακέτο να κατευθύνεται μέσω μίας ελάχιστης διαδρομής (με βάση τους περιορισμούς της τοπολογίας), ενώ ο περιορισμός του εύρους ζώνης κάθε συνδέσμου δεν παραβιάζεται, μπορούμε να εγγυηθούμε ότι η κατανάλωση ενέργειας διατηρείται όσο το δυνατόν χαμηλότερη.

3.3 Αλγόριθμος Fault-Tolerant σε NoC

Ο αλγόριθμος FaulToleReR είναι ικανός να αναδιατάξει τα πακέτα τόσο αλλάζοντας την τοπολογία όσο και με την αλλαγή της δρομολόγησης μέσω μιας διαδικασίας δρομολόγησης δύο σταδίων. Στο πρώτο στάδιο βρίσκει όλες τις διαθέσιμες διαδρομές για κάθε ροή του CTG. Αυτές οι προκαταρκτικές διαδρομές χρησιμοποιούνται στο δεύτερο βήμα για να βρεθούν οι τελικές δυνατές διαδρομές για κάθε ζεύγος πηγής-προορισμού, προκειμένου να ελαχιστοποιηθεί η εξίσωση (2). Το FaulToleReR δεν χρειάζεται εικονικό

κανάλι για χειρισμό αδιεξόδων, αλλά μπορεί να χρησιμοποιήσει εικονικά κανάλια για τη βελτίωση της απόδοσης. Ο αλγόριθμός του περιγράφεται στην εικόνα 3.1.

The **FaulToleReR** Reconfigurable Fault-Tolerant Routing Algorithms

```
1:   boolean finish = false;
2:   int dis_tsh, l_tsh;
3:   function Initionation (flows, dis_tsh) {
4:       Sort (flows); //According to a priority criteria
5:       all_paths = Find_All_Paths (flows, dis_tsh);
6:   }// end function
7:   function Main (flows, all_paths, faults) {
8:       if (faults) {
9:           Prune_All_Paths (all_paths, faults);
10:      }//enf if
11:      while (!finish) {
12:          minimals = Get_Minimal_Paths (flows, all_paths);
13:          finish = Path_Finder (flows, all_paths, minimals, l_tsh);
14:      }//end while
15:  }// end function
16:  function Path_Finder (flows, all_paths, minimals, l_tsh) {
17:      for all possible paths from all_paths {
18:          if (Link_Load (paths, l_tsh)) {
19:              if (DeadLock_Free (paths)) {
20:                  return true;
21:              }//end if
22:          }//end if
23:      }//end for
24:   }//end function
```

Εικόνα 3.1 Ψευδοκώδικας αλγορίθμου FaulToleReR

Όταν ξεκινά το σύστημα αλλάζει και η δρομολόγηση του NoC που αντιπροσωπεύεται από το CTG. Αυτή η συνάρτηση αποτελείται από δύο υπολειτουργίες, την Sort και την Find_All_Paths. Η λειτουργία Sort είναι μια τροποποιημένη έκδοση του bubble sort που ταξινομεί τις ροές σύμφωνα με

κάποια προκαθορισμένη προτεραιότητα. Καθώς στοχεύουμε στη βελτιστοποίηση της ισχύος και την μείωση της καθυστέρησης, τα κριτήρια προτεραιότητας αποτελούν και τον όγκο επικοινωνίας των ροών. Ως εκ τούτου, οι ροές δεδομένων που στοχεύουν στην μείωση της καθυστέρησης και και την βελτιστοποίηση της ισχύος έχουν μεγαλύτερη προτεραιότητα. Αυτές οι ταξινομημένες ροές και το dis_tsh, μεταφέρονται στη συνάρτηση Find_All_Paths σαν είσοδος, για να βρουν όλες τις δυνατές διαδρομές για κάθε ροή που είναι μικρότερες από τις dis_tsh (όσον αφορά τον αριθμό hops). Αυτή η συνάρτηση είναι μια προσαρμοσμένη έκδοση του αλγορίθμου Dijkstra [10].

Ο προσαρμοσμένος αλγόριθμος μειώνει σημαντικά τους υπολογισμούς του αλγορίθμου και είναι πλέον κατάλληλος για online εκτέλεση. Η έξοδος αυτής της υποσυνάρτησης είναι ένα σύνολο κατακερματισμένων διαδρομών. Μόλις γίνει αυτό το βήμα, αρχίζει το βασικό βήμα του αλγορίθμου.

Μετά την αρχική διαδικασία, ο αλγόριθμος εισέρχεται στην λειτουργία main, για να βρει μια κατάλληλη διαδρομή βασισμένη στην τρέχουσα διαμόρφωση δικτύου. Η λειτουργία αυτή ενεργοποιείται είτε από την εκκίνηση του συστήματος είτε μετά από την εμφάνιση σφάλματος. Πρώτον, ελέγχει για αλλαγή τοπολογίας και την εμφάνιση σφάλματος. Εάν δεν είναι η πρώτη φορά που αυτή η λειτουργία ξεκινά, το σύνολο των διαδρομών που έχουν καταχωρηθεί στο ευρετήριο θα πρέπει να επαναπροσδιορίζεται αφαιρώντας τις διαδρομές που περιέχουν τους ελαττωματικούς συνδέσμους. Η διαδρομή για κάθε ροή θα πρέπει να ικανοποιεί τις ακόλουθες συνθήκες:

1. Το σύνολο όλων των διαδρομών που βρέθηκαν θα πρέπει να ελαχιστοποιήσει την εξίσωση (2) προκειμένου να επιτευχθεί η ελάχιστη μέση λανθάνουσα κατάσταση σε όλες τις ροές επικοινωνίας. Η ελάχιστη τιμή της εξίσωσης (2) προκύπτει όταν κάθε πακέτο φτάσει στον προορισμό του μέσω της μικρότερης διαδρομής. Ωστόσο, αυτό μπορεί να μην είναι δυνατό λόγω

περιορισμών στους συνδέσμους, εμφάνισης αδιεξόδου ή των περιορισμών που επιβάλλονται από την εξίσωση (1). Ως εκ τούτου, μερικές από τις ροές πρέπει να παρακάμψουν το μικρότερο μονοπάτι.

Αν η μικρότερη διαδρομή μιας ροής και η πραγματική διαδρομή που εκχωρείται σε αυτήν αντιπροσωπεύεται από την route*(i, j) και τη route(i, j), αντίστοιχα, ορίζουμε την παράκαμψη της ροής ως εξής:

$$d_{f_{i,j}} = route(i, j) - route^{*}(i, j) \quad (3)$$

Δεδομένου ότι η συνολική παράκαμψη είναι η διαφορά μεταξύ του ελάχιστου μέσου μήκους διαδρομής και του μέσου μήκους διαδρομής που αποκτάται από την πραγματική δρομολόγηση, η ελαχιστοποίηση της εξίσωσης (2) ισοδυναμεί με την ελαχιστοποίηση της αθροιστικής επιβάρυνσης των ροών και ορίζετε ως εξής:

$$D = \sum_{\forall e_{i,j} \in E} d_{f_{i,j}} \quad (4)$$

2. Η συνολική κίνηση που περνά μέσω ενός συνδέσμου δεν πρέπει να υπερβαίνει το μέγιστο διαθέσιμο εύρος ζώνης για να αποφευχθεί η συμφόρηση (Εξίσωση 1). Αυτός ο περιορισμός οδηγεί σε καλύτερη απόδοση του NoC και αυξάνει την ανοχή του δικτύου από σφάλματα. Εάν ο συνολικός όγκος επικοινωνίας δύο ή περισσοτέρων ροών που αντιστοιχίζονται στον ίδιο σύνδεσμο παραβιάζουν αυτόν τον περιορισμό, δημιουργεί πρόβλημα στο δίκτυο που ονομάζεται overlapping occurs.

3. Η διαμόρφωση δρομολόγησης πρέπει να είναι deadlock και livelock free. Πολλοί αλγόριθμοι δεν εμπλέκονται με πρόβλημα αδιεξόδου και χρησιμοποιούν εικονικά κανάλια για να αποφύγουν τα αδιέξοδα. Παρόλα αυτά, το FaulToleReR αποφεύγει τα αδιέξοδα κατά την επιλογή της

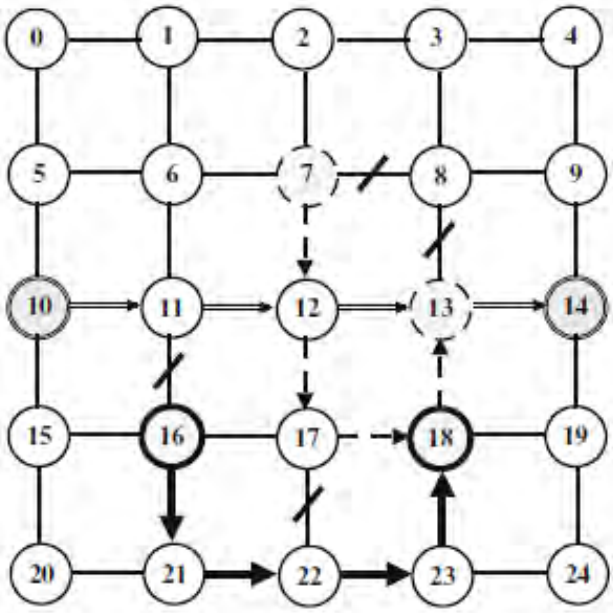
διαδρομής για να χαλαρώσει τον περιορισμό του VC που χρησιμοποιούν πολλοί αλγόριθμοι δρομολόγησης.

Στον αλγόριθμό μας, η συνάρτηση Get_Minimal_Paths, η οποία είναι η συνάρτηση που επεμβαίνει στο ευρετήριο των διαδρομών του αλγορίθμου FaulToleReR, στοχεύει στην ικανοποίηση του πρώτου αναφερόμενου παραπάνω. Αυτή η συνάρτηση ξεκινά από την ελάχιστη δυνατή τιμή για το D και επιστρέφει το σύνολο των διαδρομών με τις οποίες επιτυγχάνεται αυτή η τιμή (Η ελάχιστη τιμή είναι 0 όταν κάθε ροή κατευθύνεται μέσω της μικρότερης διαδρομής). Αν το σύνολο επιλεγμένων διαδρομών "εγκριθεί" από τις λειτουργίες Link_Load και Deadlock_Free, οι οποίες ελέγχουν εάν η ελάχιστη διαδρομή μεταξύ κόμβου προέλευσης και προορισμού ικανοποιεί τη δεύτερη και την τρίτη συνθήκη, αντίστοιχα, ο στόχος επιτυγχάνεται. Διαφορετικά, οι παράμετροι Get_Minimal_Paths αυξάνουν το D κατά μία μονάδα και το μεταβιβάζουν στη λειτουργία Path_Finder για περαιτέρω ελέγχους. Η διαδικασία αυτή επαναλαμβάνεται έως ότου το FaulToleReR εντοπίσει ένα σύνολο διαδρομών που πληρούν τις προαναφερόμενες συνθήκες.

Η λειτουργία Get_Minimal_Paths χρησιμοποιεί δύο προκαθορισμένες δομές δεδομένων, τη μνήμη και την visited. Αρχικά, η συνάρτηση κάνει την υπόθεση ότι κάθε ροή θα έχει τη μικρότερη δυνατή διαδρομή από την πηγή στον προορισμό, θέτοντας το συνολικό overhead παρακάμψεως του δικτύου ίσο με το μηδέν. Στη συνέχεια, στέλνει την n-tuple στη λειτουργία Path_Finder για να ελέγξει εάν πληροί τα κριτήρια της ικανότητας σύνδεσης και της αποφυγής του αδιεξόδου. Εάν οι έλεγχοι δείξουν ότι δεν παραβιάζετε κάποιος περιορισμός, ο αλγόριθμος ενημερώνει τους σχετικούς πίνακες δρομολόγησης και τερματίζει. Διαφορετικά, ο αλγόριθμος επιστρέφει στη λειτουργία Get_Minimal_Paths για να βρει άλλες διαδρομές.

Όταν ο αλγόριθμος επιστρέφει για δεύτερη φορά στο Get_Minimal_Paths, επιλέγει την τελευταία ροή της ταξινομημένης λίστας (η οποία θα είναι η ροή με το χαμηλότερο κόστος) και εκχωρεί στη διαδρομή ένα βήμα που κάνει τη διαδρομή μεγαλύτερη από το προηγούμενο μήκος της. Για παράδειγμα, εάν η συντομότερη διαδρομή από την πηγή προς τον προορισμό για αυτή τη ροή είναι 5, μπορεί να φτάσει στον προορισμό μετακινώντας διαδρομές μήκους 5, 7, 9, 11 κ.λπ. Έτσι, το μήκος της επόμενης μικρότερης διαδρομής αυτής είναι 7. Το μήκος των άλλων ροών παραμένει το ίδιο. Το νέο σύνολο διαδρομών έχει το ελάχιστο overhead μεγαλύτερο από 0. Και πάλι, το FaulToleReR αποθηκεύει την n-tuple που επισκέπτεται και τα στέλνει στη λειτουργία Path_Finder για να ελέγξει αν είναι η τελική λύση. Σε πολλές εφαρμογές, όπου υπάρχουν πολλές ροές και είναι πολύ πιθανό η επικάλυψη διαδρομής να επιφέρει υπέρβαση του ορίου ικανότητας σύνδεσης, αυτή η λειτουργία μπορεί να χρησιμοποιηθεί περισσότερο από δύο φορές.

Στο επόμενο βήμα, ο αλγόριθμος εκχωρεί την επόμενη πιο σύντομη διαδρομή στο δεύτερο βήμα με την αυξανόμενη σειρά του όγκου επικοινωνίας, ενώ όλες οι άλλες ροές έχουν την αρχική συντομότερη διαδρομή. Μετά τον υπολογισμό της συνολικής παράκαμψης και την αποθήκευση του στη δομή δεδομένων Visited, ο αλγόριθμος συνδυάζει αυτό το σύνολο με όλα τα άλλα σύνολα με διαφορετικές τιμές για την ίδια ροή, που αποθηκεύονται στο Visited για να δημιουργήσουν ένα νέο σετ δεδομένων. Το αποτέλεσμα είναι μια νέα n-tuple, η οποία είναι το άθροισμα των παλαιών παρακάμψεων. Το FaulToleReR αποθηκεύει αυτό το σύνολο n-tuple στη δομή δεδομένων μνήμης. Αυτή η διαδικασία επαναλαμβάνεται και αν υπάρχει ένα σύνολο στη δομή δεδομένων μνήμης που έχει χαμηλότερη παράκαμψη από το νέο, ο αλγόριθμος το επιλέγει.



The *Visited* data structure

Path Length (f1,f2,f3)	Detour Over-head
(4,2,2)	0
(4,2,4)	100
(4,4,2)	160
(6,2,2)	180
(4,2,6)	200
(4,6,2)*	320

The *Memory* data structure

Path Length (f1,f2,f3)	Detour Over-head
(4,4,4)*	260
(6,2,4)	280
(6,4,2)	340

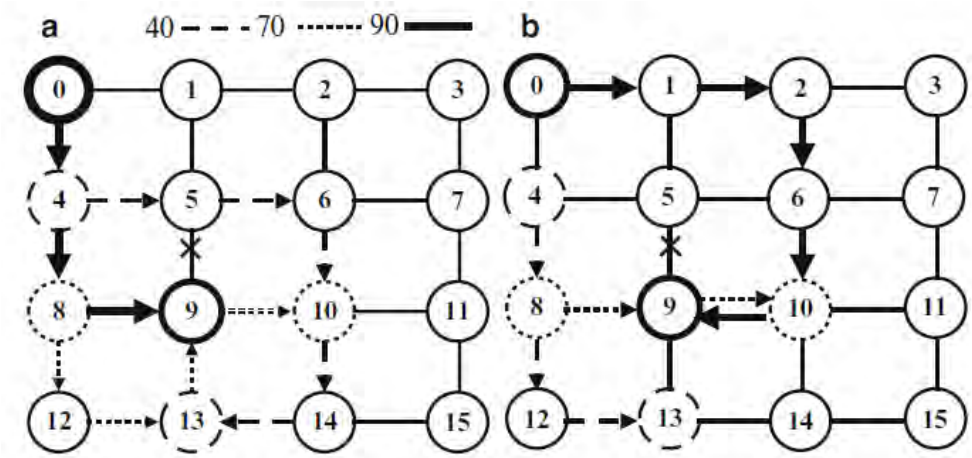
Εικόνα 3.2 Παράδειγμα εκτέλεσης της συνάρτησης Get_Minimal_Paths

Παίρνουμε ως παράδειγμα ένα NoC όπως αυτό απεικονίζεται στην εικόνα 3.2 με τέσσερις ελαττωματικούς συνδέσμους και μέγιστο εύρος ζώνης σύνδεσης 100 Mbps. Υπάρχουν τρεις ροές με διαφορετικούς όγκους επικοινωνίας. Το Flow1 συνδέει τον κόμβο 7 με τον κόμβο 13 με όγκο επικοινωνίας 50 Mbps. Το Flow2 συνδέει τον κόμβο 16 με τον κόμβο 18 με τον όγκο επικοινωνίας 80 Mbps και το Flow3 συνδέει τον κόμβο 10 με τον κόμβο 14 με όγκο

επικοινωνίας 90 Mbps. Ο αλγόριθμος επαναλαμβάνει έξι φορές για να βρει την καλύτερη λύση της δεδομένης διαμόρφωσης. Τα σύνολα και το detour overheads για κάθε επανάληψη φαίνονται στο σχήμα.

Υποθέτουμε ότι οι τελικές διαδρομές που βρέθηκαν αποδίδουν χαμηλότερη μέση λανθάνουσα κατάσταση σε σύγκριση με άλλες σχετικές μεθόδους, επειδή ο αλγόριθμος FaulToleReR στοχεύει στην ελαχιστοποίηση της εξίσωσης 4. Επιπλέον, ο προτεινόμενος αλγόριθμος εγγυάται την απουσία αδιεξόδου χωρίς τη χρήση εικονικών καναλιών, αλλά με τη λήψη κάποιων αποτρεπτικών κανόνων που περιορίζουν τις διαδρομές από την πηγή στον προορισμό. Αυτό επιτυγχάνεται με την κατασκευή ενός διαγράμματος εξάρτησης καναλιού, CDG, των επιλεγμένων διαδρομών και ελέγχει αν η διαδρομή μας κάνει κύκλο ή όχι. Σε περίπτωση που δεν βρεθούν κύκλοι, η διαμόρφωση δρομολόγησης είναι σίγουρα deadlock free.

Το FaulToleReR είναι deadlock free. Με άλλα λόγια, σίγουρα βρίσκει μια διαδρομή για κάθε ροή προς τον προορισμό της, αρκεί το δίκτυο να μπορεί να την υποστηρίξει. Όπως αναφέρθηκε προηγουμένως, τοποθετούμε έναν περιορισμό χρήσης του εύρους ζώνης σε κάθε σύνδεσμο για να αποφύγουμε τον κορεσμό του δικτύου. Συνεπώς, επεκτείνουμε την έννοια της αποσύνδεσης, στην οποία δύο κόμβοι αποσυνδέονται όχι μόνο όταν δεν υπάρχουν φυσικές συνδέσεις μεταξύ τους, αλλά και όταν χρησιμοποιούμε πιθανές διαδρομές μεταξύ τους μπορεί να οδηγήσουν σε παραβίαση του περιορισμού του εύρους ζώνης ορισμένων συνδέσμων.



Εικόνα 3.3 Ο αντίκτυπος της στην επιβάρυνση της παράκαμψης στην προτεινόμενη δρομολόγηση τριών ροών

Ο προτεινόμενος αλγόριθμος μας εφαρμόζει ένα είδος διαδρομής, έτσι ώστε οι ροές με χαμηλότερη προτεραιότητα να μην παραγκωνίζονται πάντα από εκείνες με υψηλότερη προτεραιότητα. Δυστυχώς, το FaulToleReR φαίνεται να επιλέγει πρωτίστως να παρακάμψει μια ροή με χαμηλότερο όγκο επικοινωνίας υπέρ μιας αλληλεπικαλυπτόμενης ροής με υψηλότερο όγκο επικοινωνίας. Ωστόσο, δεν είναι η καλύτερη στρατηγική σε όλες τις περιπτώσεις. Για παράδειγμα, υποθέστε ότι μια διαμόρφωση δικτύου που φαίνεται στην εικόνα 3.3. Υπάρχουν τρεις ροές με όγκο επικοινωνίας 90, 70 και 40Mbps, ενώ το μέγιστο εύρος ζώνης σύνδεσης είναι 100 Mbps. Καθώς υπάρχουν απλοί σύνδεσμοι μεταξύ των μικρότερων διαδρομών και το συνολικό φορτίο συνδέσμου δεν μπορεί να υπερβεί τον περιορισμό του εύρους ζώνης, ορισμένες ροές θα πρέπει να παρακάμπτουν τις μικρότερες διαδρομές τους. Η εικόνα 3.3.α απεικονίζει μια περίπτωση όπου δύο χαμηλότερες ροές παρακάμπτουν μια από τις μικρότερες διαδρομές τους για να εξασφαλίσουν μια συντομότερη διαδρομή για τη ροή με υψηλότερο όγκο. Συνεπώς, η συνολική υπέρβαση της παρακάμψεως είναι 200 (140 για τη ροή των 70 Mbps και 80 για τη ροή των 40 Mbps και σύμφωνα με την εξίσωση 4). Από την άλλη πλευρά, η διαμόρφωση δρομολόγησης που φαίνεται στην

εικόνα 3.3.β, το οποίο έχει ληφθεί από την λανθασμένη εκροή της μεγαλύτερης ροής, έχει συνολική υπέρβαση της παρακάμψεως 180, προφανώς προτιμητέα από την ολική επιβάρυνση της παράκαμψης της εικόνας 3.3.α.

3.4 Πειραματικά αποτελέσματα

3.4.1 Περιβάλλον προσομοίωσης

Ο προτεινόμενος μηχανισμός δρομολόγησης είναι ανθεκτικός σε σφάλματα και αξιολογείται βάσει ορισμένων ρεαλιστικών σημείων αναφοράς. Το σετ αναφοράς περιλαμβάνει ορισμένα διαγράμματα εργασιών μαζί με κάποιες υπάρχουσες εφαρμογές που έχουν χρησιμοποιηθεί ευρέως, όπως το Multimedia Media System (MMS) [11] και ο αποκωδικοποιητής Global System for Mobile Communications (GSM) [12].

Η αξιολόγηση του αλγορίθμου FaulToleReR γίνετε μέσα από τον BookSim, έναν εξομοιωτή δικτύων διασύνδεσης με ακρίβεια κύκλου [13]. Χρησιμοποιήθηκε η βιβλιοθήκη PowerSim by Orion2.0 [14] προκειμένου να αξιολογήσουμε την κατανάλωση ισχύος των NoCs με διαφορετικούς αλγορίθμους δρομολόγησης. Τα αποτελέσματα ισχύος βασίζονται σε τεχνολογία NoC 128 bit που εφαρμόζεται σε τεχνολογία 65 nm και η συχνότητα ποικίλλει ανάλογα με τις διαφορετικές διαστάσεις του δικτύου. Η τοποθέτηση σφάλματος στο δίκτυο γίνεται με τυχαία απενεργοποίηση ορισμένων συνδέσμων με βάση ένα δεδομένο ποσοστό σφαλμάτων. Ένας δρομολογητής καθίσταται μη λειτουργικός όταν όλες οι συνδέσεις που συνδέονται με αυτόν καθίστανται ελαττωματικές. Το FaulToleReR, όπως αναφέρθηκε προηγουμένως, μπορεί να ανεχτεί τόσα λάθη όσα αναφέρθηκαν στην προηγούμενη ενότητα του συγκεκριμένου κεφαλαίου.

Το μέγεθος του NoC στον προσομοιωτή μας είναι 4×4, 6×6 και 8×8. Η τοπολογία NoC είναι ένα συμβατικό πλέγμα 2D και ο NoC υιοθετεί την

εναλλαγή wormhole με πακέτα 4-flit και 16 buffer για κάθε εικονικό κανάλι. Η διάρκεια ζωής προσομοίωσης είναι 6.000.000 κύκλοι για τον realistic προσομοιωτή και 10.000.000 κύκλους για τον synthetic προσομοιωτή .

Επιλέγουμε δύο αλγορίθμους δρομολόγησης για σύγκριση : τον αλγόριθμο δρομολόγησης planar [15] και τον Highly Resilient Routing Algorithm (HRA) [16]. Ο Planar είναι ένας γνωστός αλγόριθμος προσαρμοστικής δρομολόγησης που χρησιμοποιεί τοπικές πληροφορίες για την προσαρμογή των πακέτων. Χρειάζεται ορισμένα εικονικά κανάλια να λειτουργούν σωστά και θέτει ορισμένους περιορισμούς στην κατανομή των εικονικών καναλιών, προκειμένου να διασφαλιστεί η ελευθερία του αδιεξόδου. Ο HRA είναι ένας υπερσύγχρονος κατανεμημένος αλγόριθμος δρομολόγησης ανθεκτικός στα σφάλματα, ο οποίος επαναπροσδιορίζει το NoC για την αποφυγή ελαττωματικών συνδέσμων. Ο HRA απαγορεύει ορισμένους συνδέσμους και στροφές προκειμένου να αποφευχθεί το αδιέξοδο.

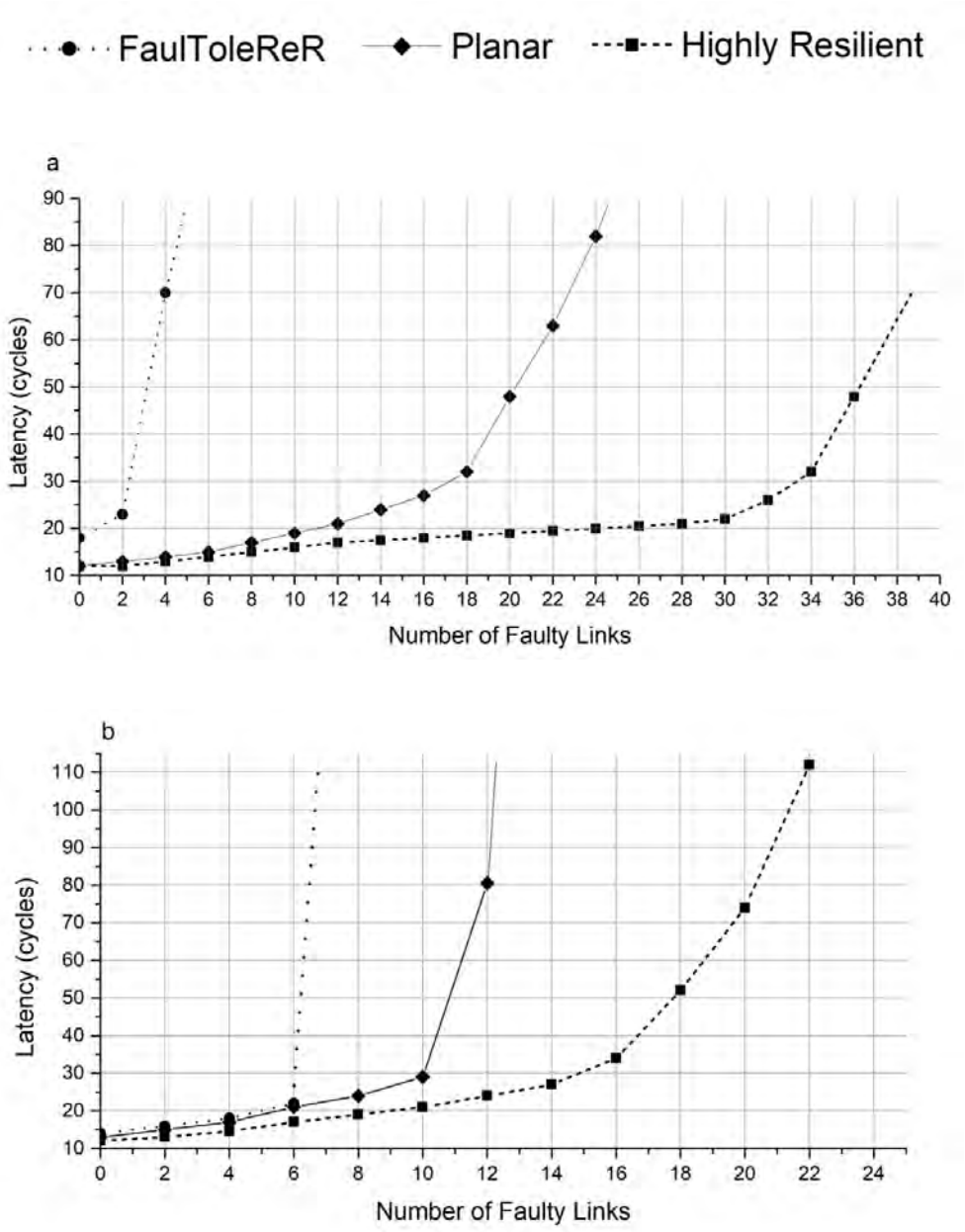
3.4.2 Μέση καθυστέρηση μηνύματος

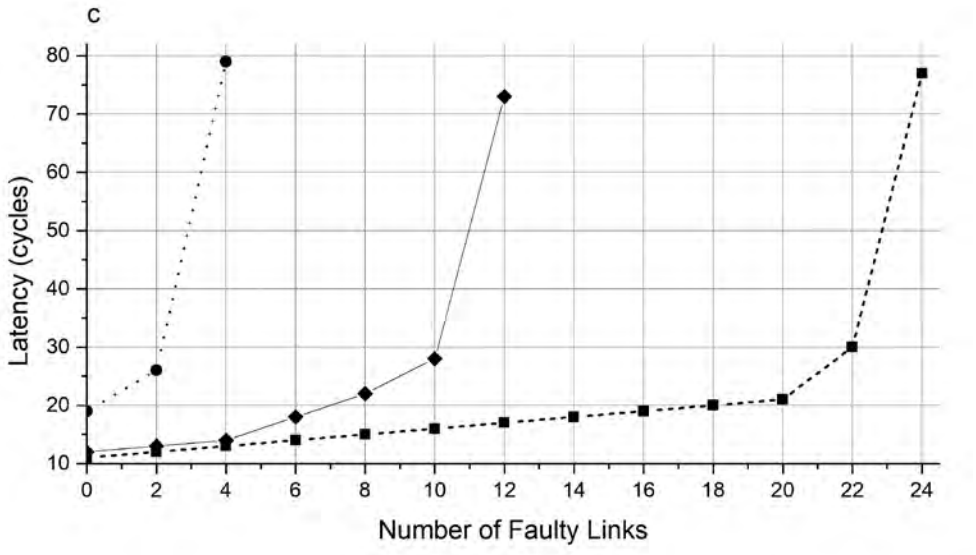
Στην πρώτη περίπτωση, συγκρίνουμε τη μέση καθυστέρηση μηνύματος (Average Message Latency - AML) του FaulToleReR με άλλους αλγόριθμους δρομολόγησης που εξετάστηκαν κάτω από διαφορετικό αριθμό τυχαίων injected faults στους συνδέσμους. Στην εικόνα 3.4 εμφανίζονται οι μέσες καθυστερήσεις μηνυμάτων που δίνονται από τα τρία προγράμματα δρομολόγησης κάτω από διαφορετικά σημεία αναφοράς. Από τα αποτελέσματα προσομοίωσης που απεικονίζονται στην εικόνα 3.4 συμπεραίνουμε ότι:

- Το FaulToleReR και το planar έχουν την ίδια συμπεριφορά όταν δεν υπάρχουν σφάλματα. Ωστόσο, η AML της HRA είναι λίγο υψηλότερη, λόγω των ειδικών κανόνων δρομολόγησης που απαγορεύουν τη χρήση ορισμένων συνδέσμων και περιστροφών προκειμένου να αποφευχθεί το deadlock.

Συνεπώς, ο αριθμός των διαθέσιμων διαδρομών μεταξύ δύο δεδομένων κόμβων προέλευσης και προορισμού είναι μικρότερος από τον αριθμό του planar και του FaulToleReR που δεν έχουν αυτούς τους περιορισμούς.

- Όσο αυξάνεται ο αριθμός των ελαττωματικών συνδέσεων, τόσο αυξάνεται και η AML όλων των αλγορίθμων δρομολόγησης. Ο λόγος είναι ότι οι ελαττωματικοί σύνδεσμοι μειώνουν τους πόρους του δικτύου και το φορτίο των ελαττωματικών συνδέσεων κατανέμεται μεταξύ άλλων συνδέσμων, οδηγώντας σε αύξηση του φόρτου και, συνεπώς, την καθυστέρηση. Και στις τρεις περιπτώσεις που απεικονίζονται στην εικόνα 4.4, ο προτεινόμενος αλγόριθμος μας έχει λιγότερη AML σε σύγκριση με τους άλλους αλγόριθμους δρομολόγησης για το πλήρες φάσμα του θεωρούμενου αριθμού βλαβών (αριθμός ελαττωματικών συνδέσμων). Αυτό οφείλεται στο γεγονός ότι ο αλγόριθμος μας βρίσκει πάντα την καλύτερη διαδρομή για τα πακέτα με βάση την τρέχουσα τοπολογία και το πρότυπο επικοινωνίας. Επιπλέον, ο αλγόριθμος FaulToleReR μετατοπίζει το σημείο κορεσμού δικτύου ώστε να ανεχθεί περισσότερα σφάλματα (όταν η κίνηση είναι σταθερή και η καθυστέρηση αυξάνεται λόγω ελαττωματικών συνδέσεων) και ο κορεσμός του δικτύου μετατοπίζεται σε υψηλότερα ποσοστά fault injection, σε σύγκριση με τους άλλους αλγορίθμους δρομολόγησης.





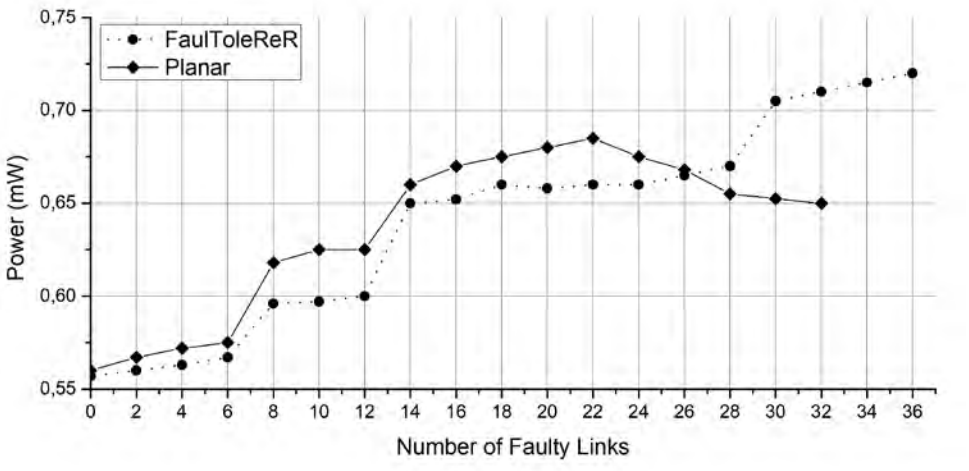
Εικόνα 4.4 Σύγκριση μέσης λανθάνουσας σχέσης μεταξύ FaulToleReR, Planar και ενός εξαιρετικά ανθεκτικού αλγορίθμου (highly resilient algorithm) για δεδομένο αριθμό faults links (α) GSM, (β) MMS και (γ) uniform input traffic

3.4.3 Ανάλυση ισχύος

Συγκρίνουμε περαιτέρω τον αλγόριθμο FaulToleReR με τον planar ως προς την κατανάλωση ενέργειας. Καθώς το FaulToleReR προωθεί πακέτα μέσω βραχύτερων διαδρομών από τον planar, τα πακέτα περνούν από λιγότερους ενδιάμεσους δρομολογητές. Ως εκ τούτου, αναμένουμε από τον προτεινόμενο αλγόριθμό μας να καταναλώνει λιγότερη μέση ισχύ από τους άλλους. Η συνολική ισχύς που καταναλώνεται από ένα πακέτο είναι το άθροισμα της ισχύος που καταναλώνει σε κάθε κόμβο: εγγραφή / ανάγνωση του πακέτου προς / από το buffer, διαβάθμιση διασταύρωσης, link traversal, υπολογισμός διαδρομής, virtual channel arbitration, και switch arbitration.

Στην εικόνα 5 φαίνεται η κατανάλωση ισχύος των αλγορίθμων δρομολόγησης FaulToleReR και Planar σε ένα δίκτυο NoC 6×6 υπό την κίνηση GSM. Η εικόνα δείχνει ότι το FaulToleReR έχει μικρότερη κατανάλωση ενέργειας

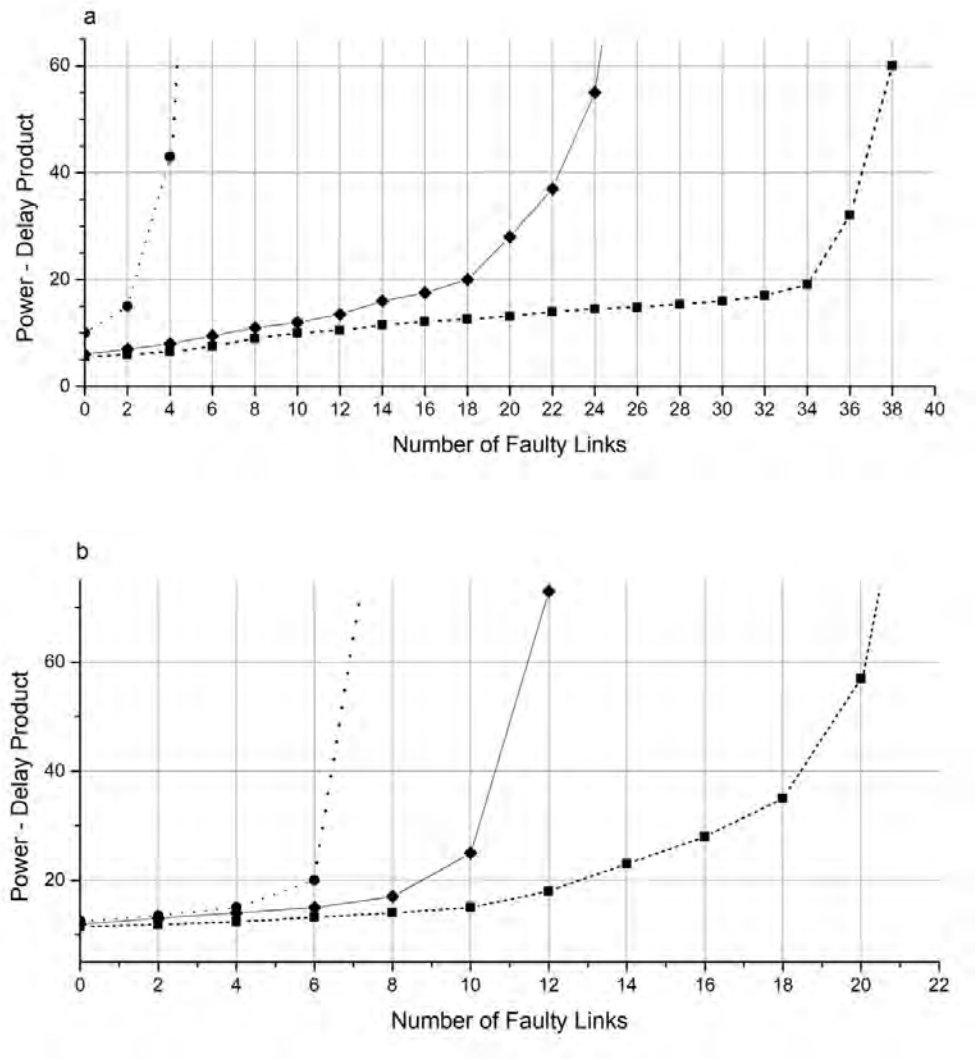
όταν υπάρχουν λιγότεροι από 27 ελαττωματικοί σύνδεσμοι. Όπως φαίνεται και στην εικόνα 4.4.α δείχνει ότι το δίκτυο εισέρχεται στην κατάσταση κορεσμού με τον αλγόριθμο δρομολόγησης planar παρουσία 22 και περισσότερων βλαβών. Από αυτό το σημείο, το FaulToleReR καταναλώνει περισσότερη ισχύ, καθώς εξακολουθεί να λειτουργεί κανονικά, ενώ κάτω από τους άλλους αλγόριθμους, ο ρυθμός αποδοχής της κυκλοφορίας NoC μειώνεται σημαντικά, οδηγώντας στον περιορισμό της κατανάλωσης ενέργειας του NoC. Ως αποτέλεσμα, η κατανάλωση ενέργειας του FaulToleReR είναι υψηλότερη από τους άλλους αλγόριθμους μετά το σημείο κορεσμού.

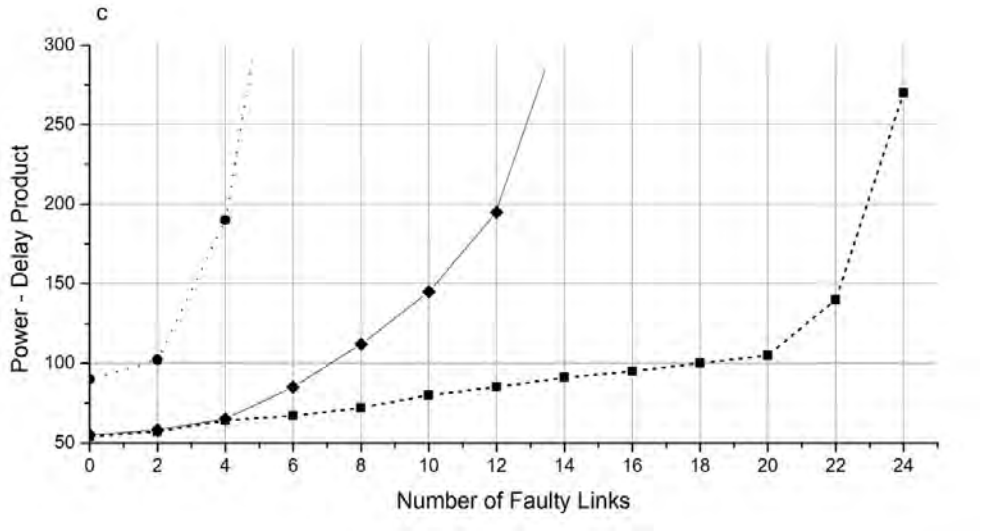


Εικόνα 3.5 Σύγκριση κατανάλωσης ενέργειας μεταξύ FaulToleReR και planar

Στη συνέχεια, υπολογίζουμε την καθυστέρησης ισχύος (PDP) των τριών NoCs για να κατανοήσουμε καλύτερα τη συμπεριφορά των αλγορίθμων δρομολόγησης κατά την αντιμετώπιση βλαβών. Η εικόνα 3.5 δείχνει την PDP για έναν δεδομένο αριθμό βλαβών και κυκλοφορίας για τα FaulToleReR, Planar και HRA. Ακολουθώντας την ίδια τάση με τα αποτελέσματα λανθάνουσας κατάστασης, το PDP του FaulToleReR ξεπερνά τους άλλους αλγόριθμους που εξετάστηκαν κάτω από όλα τα σημεία αναφοράς. Στην

εικόνα 3.6 έχουμε αποκλείσει τα αποτελέσματα της φόρμας στατικής ισχύος, καθώς δεν επηρεάζεται από τον αλγόριθμο δρομολόγησης.





Εικόνα 3.6 Σύγκριση αλγορίθμων FaulToleReR, Planar και HRA για ένα δεδομένο αριθμό βλαβών (α) με GSM, (β) με MMS και (γ) με uniform input traffic

3.5 Συμπεράσματα

Το FaulToleReR είναι ένα κεντρικό σύστημα δρομολόγησης, με στόχο να αντιμετωπίστουν οι δυναμικές αλλαγές της τοπολογίας στα NoCs που προκαλούνται από μόνιμες και διαλείπουσες βλάβες. Πρόκειται για ένα σημαντικό κεφάλαιο στις νανοτεχνολογίες, καθώς τα συστήματα on-chip γίνονται ολοένα και πιο ευαίσθητα σε σφάλματα. Ολόκληρη η διαδικασία βασίζεται στην παρακολούθηση της κατάστασης των στοιχείων NoC (για να εξακριβωθεί αν είναι ελαττωματικά ή όχι), καθώς επίσης και στο μοντέλο κίνησης των chip και την αλλαγή της διαδρομής των πακέτων σε ανταπόκριση σε μια αλλαγή στην τοπολογία NoC (που προκαλείται από την αφαίρεση faulty links). Χρησιμοποιώντας προσέγγιση κεντρικής διαχείρισης, αποδείξαμε ότι ο συγκεκριμένος αλγόριθμος μας μπορεί να μειώσει αποτελεσματικά την ισχύ και τις επιδόσεων σε σχέση με άλλους αλγόριθμους δρομολόγησης.

Βιβλιογραφία

1. A. Alaghi, M. Sedghi, N. Karimi, M. Fathy, Z. Navabi, Reliable NoC architecture utilizing a robust rerouting algorithm, in *Proceedings of the 6th IEEE East–West Design and Test Symp. (EWDTS)* (Kharkov, Poland, 2008), pp. 101–105

2. A. Patooghy, S.G. Miremadi, XYX: A power & performance efficient fault-tolerant routing algorithm for network on chip, in *Proceedings of the 17th EuroMicro International Conference on Parallel, Distributed and Networks-Based Processing* (Weimar, Germany, 2009), pp. 245–251

3. S. Pasricha, N. Dutt, *On-Chip Communication Architectures* (Morgan Kauffman, Boston, 2008)

4. C. Constantinescu, Intermittent faults in VLSI circuits, *in Proceedings of IEEE Workshop on System Effects of Logic Sot Errors*, Urbana-Champaign, USA, 2006

5. A.H. Ajami, K. Banjerjee, M. Pedram, Modeling and analysis of nonuniform substrate temperature effects on global VLSI interconnects. IEEE Tans. Comp. Aid. Des. Integ. Circuit. Syst. **24**(6), 849–861 (2005)

6. K. Banerjee, A. Mehrotra, A.S. Vincentelli, C. Hu, On thermal effects in deep sub-micron VLSI interconnects, in *Proceedings of the Design Automation Conference (DAC)* (New Orleans, LA, USA, 1999), pp. 885–891

7. J. Hu, R.Marculescu, Exploiting the routing flexibility for energy/performance aware mapping of regular NoC architectures, in *Proceedings of Design, Automation, and Test in Europe Conference* (Munich, Germany, 2003), pp. 668–693

8. J. Duato, A necessary and sufficient condition for deadlock-free adaptive routing in wormhole networks. IEEE Trans. Parall. Distrib. Syst. **6**(10), 1055–1067 (1995)

9. S. Lin, C. Huang, C. Chao, K. Huang, A. Wu, Traffic-balanced routing algorithm for irregular mesh-based on-chip networks. IEEE. Trans. Comp. **57**(9), 1156–1168 (2008)

10. M. Modarressi, A. Tavakkol, H. Sarbazi-Azad, VIP: Virtual point-to-point connections in NoCs. IEEE Trans. Comp. Aid. Desi. Integ. Circuit. Syst. (TCAD) **29**(6) (June, 2010)

11. J. Hu, U.Y. Ogras, R. Marculscu, System-level buffer allocation for application-specific networks-on-chip router design. IEEE Tans. Comp. Aid. Des. Integ. Circuit. Syst. **25**(12), 2919–2933 (2006)

12. M. Schmitz, Energy minimization techniques for distributed embedded systems, Ph.D. thesis, University of Southampton, 2003

13. BookSim: A cycle-accurate interconnection network simulator, Version 2.0, (2013), https://nocs.stanford.edu/cgibin/trac.cgi/wiki/Resources/BookSim

14. A.B. Kahng, B. Li, L. Peh, K. Samadi, ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration, in *Proceedings of Design, Automation, and Test in Europe Conference (DATE)* (Nice, France, 2009), pp. 423–428

15. R.V. Boppana, S. Chalasani, Fault-tolerant wormwhole routing algorithms for mesh networks. IEEE Trans. Comp. **44**(7), 848–864 (1995)

16. D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, D. Blaauw, A highly resilient routing algorithm for fault-tolerant NoCs, in *Proceedings of*

the Design, Automation, and Test in Europe (DATE) (Nice, France, 2009), pp. 21–26

$4^{ο}$ ΚΕΦΑΛΑΙΟ

ΑΞΙΟΠΙΣΤΟΙ ΚΑΙ ΠΡΟΣΑΡΜΟΣΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ ΔΡΟΜΟΛΟΓΗΣΗΣ ΓΙΑ 2D ΚΑΙ 3D NETWORKS-ON-CHIP

4.1 Εισαγωγή

Η πρόοδος της τεχνολογίας ημιαγωγών καθιστούν δυνατή την ενσωμάτωση δισεκατομμυρίων θυρών και εκατοντάδων μονάδων επεξεργασίας σε ένα μοναδικό τσιπ [1]. Ωστόσο, οι αρχιτεκτονικές διαύλων δεν είναι αρκετά ανεπτυγμένες ώστε να παρέχουν επικοινωνία υψηλής απόδοσης μεταξύ των μονάδων επεξεργασίας. Από την άλλη πλευρά, η αξιοπιστία είναι ένα άλλο σημαντικό θέμα στους διαύλους, καθώς ένα σφάλμα μπορεί να απενεργοποιήσει ολόκληρο το σύστημα ή ένα σημαντικό μέρος του. Τα χαρακτηριστικά επεκτασιμότητας, επαναχρησιμοποίησης και αξιοπιστίας του NoC το καθιστούν σαν ένα σημαντικό υποψήφιο αντικατάστασης του κλασσικού πλέον διαύλου. Σε ένα NoC, κάθε πυρήνας συνδέεται με ένα δρομολογητή από μια διεπαφή τοπικού δικτύου. Οι πυρήνες μπορούν να επικοινωνούν μεταξύ τους με την προώθηση πακέτων μέσω δρομολογητών στο δίκτυο. Κάθε δρομολογητής συνδέεται με τους γείτονές του μέσω αμφίδρομων συνδέσεων. Το NoC επιτρέπει την επικοινωνία μεταξύ των

μονάδων επεξεργασίας μέσω διαφορετικών διαδρομών, έτσι ώστε εάν μία διαδρομή είναι ελαττωματική, μπορούν να χρησιμοποιηθούν και οι άλλες διαδρομές. Αυτό το χαρακτηριστικό καθιστά τον NoC πιο αξιόπιστο από έναν δίαυλο επικοινωνίας.

Καθώς το μέγεθος ενός δικτύου 2D κλιμακώνεται, η καθυστέρηση μετάδοσης μεταξύ απομακρυσμένων δρομολογητών αυξάνεται σημαντικά, γεγονός που έχει ως αποτέλεσμα χαμηλότερες επιδόσεις και υψηλότερη κατανάλωση ενέργειας. Επιπλέον, ο σχεδιασμός 2D IC απαιτεί μια πολύ μεγάλη περιοχή τσιπ καθώς ο αριθμός των πυρήνων αυξάνεται. Λαμβάνοντας υπόψη τα σημεία συμφόρησης του 2D, η τεχνολογία κινείται προς την έννοια των τρισδιάστατων ολοκληρωμένων κυκλωμάτων όπου πολλαπλά στρώματα ενεργού πυριτίου είναι στοιβαγμένα κάθετα. Ο συνδυασμός των πλεονεκτημάτων των συστημάτων 3D IC και των NoC παρέχουν μια σημαντική αύξηση της απόδοσης για τα SoCs. Τα επίπεδα του 3D NoC, είναι στοιβαγμένα το ένα πάνω στο άλλο και συνδέονται μέσω κατακόρυφων διασυνδέσεων που μέσω αυτών επικοινωνούν τα επίπεδα μεταξύ τους. Η συγκόλληση καλωδίων, η micro-bump (capacitive or inductive) και η through-silicon-via (TSV) είναι μερικές από τις τεχνολογίες κάθετης διασύνδεσης που έχουν χρησιμοποιηθεί σε τρισδιάστατες δομές [2,3]. Οι προσεγγίσεις διασύνδεσης TSV έχουν τη δυνατότητα να προσφέρουν τον μεγαλύτερο όγκο πληροφορίας και ως εκ τούτου είναι οι πιο αποδοτικές μεταξύ των άλλων τεχνικών κάθετης διασύνδεσης [4, 5]. Τα πλεονεκτήματα των τρισδιάστατων ολοκληρωμένων κυκλωμάτων είναι η σημαντική μείωση του μέσου μήκους διαδρομής και καθυστέρησης διάδοσης, με αποτέλεσμα τη χαμηλότερη κατανάλωση ενέργειας και την καλύτερη απόδοση [6-9]. Ωστόσο, εάν ο αριθμός των πυρήνων IP και των μνημών αυξάνεται σε κάθε στρώμα, απαιτούνται περισσότερα TSV για την αντιμετώπιση της επικοινωνίας μεταξύ επιπέδων. Δεδομένου ότι η TSV χρησιμοποιεί ένα

στρώμα για τη συγκόλληση, η επιφάνεια του chip μας αυξάνεται σημαντικά [10].

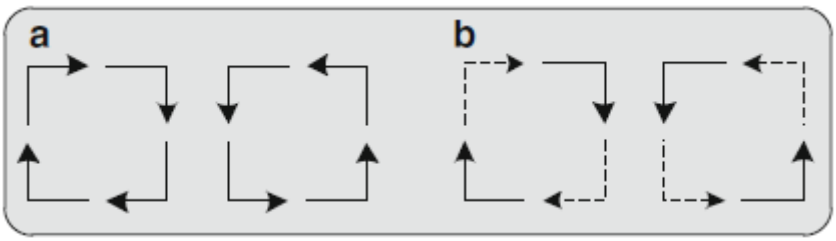
Οι αλγόριθμοι δρομολόγησης μπορούν να κατηγοριοποιηθούν ως ντετερμινιστικοί και ως προσαρμοστικοί. Η απλούστερη μέθοδος προσδιοριστικής δρομολόγησης είναι η dimension-order, η οποία είναι γνωστή ως XY και XYZ σε δίκτυα 2D και 3D, αντίστοιχα. Οι εφαρμογές ντετερμινιστικών αλγορίθμων είναι απλές, αλλά δεν είναι σε θέση να διανέμουν πακέτα και να αντιμετωπίσουν αποτελεσματικά τα προβλήματα των ελαττωματικών συνδέσμων. Στους προσαρμοστικούς αλγόριθμους, ένα πακέτο μπορεί να διασχίσει μέσω πολλαπλών διαδρομών και να μεταβεί από τον δρομολογητή προέλευσης στον δρομολογητή προορισμού. Η προσαρμοστική δρομολόγηση έχει χρησιμοποιηθεί στα δίκτυα διασύνδεσης για τη βελτίωση της απόδοσης του δικτύου και την αντοχή σε σφάλματα. Οι προσαρμοστικοί αλγόριθμοι δρομολόγησης μπορούν να είναι μερικώς ή πλήρως προσαρμοστικοί. Στους αλγορίθμους δρομολόγησης που είναι μερικώς προσαρμοστικοί, τα πακέτα περιορίζονται να επιλέξουν ανάμεσα σε μερικές ελάχιστες διαδρομές, ενώ στους πλήρως προσαρμοστικούς, επιτρέπεται στα πακέτα να παίρνουν οποιεσδήποτε από βραχύτερες διαδρομές είναι διαθέσιμες από το σύστημα [11, 12].

Οι αλγόριθμοι δρομολόγησης που είναι ανθεκτικοί σε σφάλματα σε 2D και σε 3D NoC είναι αποδοτικότεροι σε αρχιτεκτονικές mesh. Στους αλγόριθμους αυτούς χρησιμοποιούνται οι μικρότερες διαδρομές όταν η πηγή και ο προορισμός δεν βρίσκονται στην ίδια σειρά ή στήλη.

4.2 Αλγόριθμος δρομολόγησης Fault-Tolerant σε δίκτυο 2D Mesh

4.2.1 Πλήρως προσαρμοστική δρομολόγηση σε δίκτυο 2D Mesh

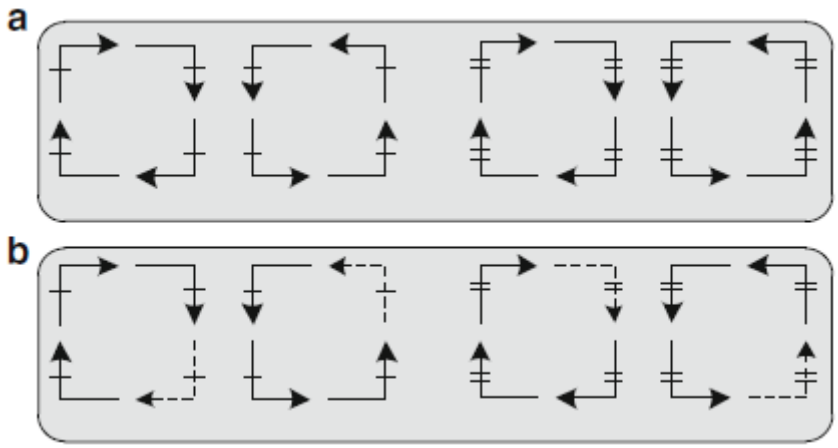
Υπάρχουν δύο τύποι πλήρων κύκλων που μπορούν να διαμορφωθούν στο δίκτυο, γνωστοί ως δεξιόστροφος και αριστερόστροφος (Εικόνα 4.1). Η δημιουργία κύκλων μπορεί να οδηγήσει σε αδιέξοδο στο δίκτυο και συνεπώς θα πρέπει να αποφευχθεί. Με τη σειρά τους, απαγορεύονται ορισμένες στροφές από κάθε κύκλο, προκειμένου να σπάσουν όλες οι κυκλικές εξαρτήσεις και έτσι να αποφευχθεί το αδιέξοδο. Στον αλγόριθμο δρομολόγησης XY, για παράδειγμα, τα πακέτα δρομολογούνται κατά μήκος της διάστασης X πριν προχωρήσουν κατά μήκος της διάστασης Y. Όπως φαίνεται στην εικόνα 1 β), σε αυτόν τον αλγόριθμο αποφεύγονται δύο στροφές από κάθε κύκλο και έτσι δεν υπάρχει δυνατότητα σχηματισμού ενός πλήρους κύκλου μεταξύ των υπόλοιπων στροφών.



Εικόνα 4.1 α) Δεξιόστροφες και αριστερόστροφες στροφές. β) επιτρεπόμενες και απαγορευμένες στροφές στον αλγόριθμο δρομολόγησης XY (οι διακεκομμένες γραμμές δείχνουν τις απαγορευμένες στροφές)

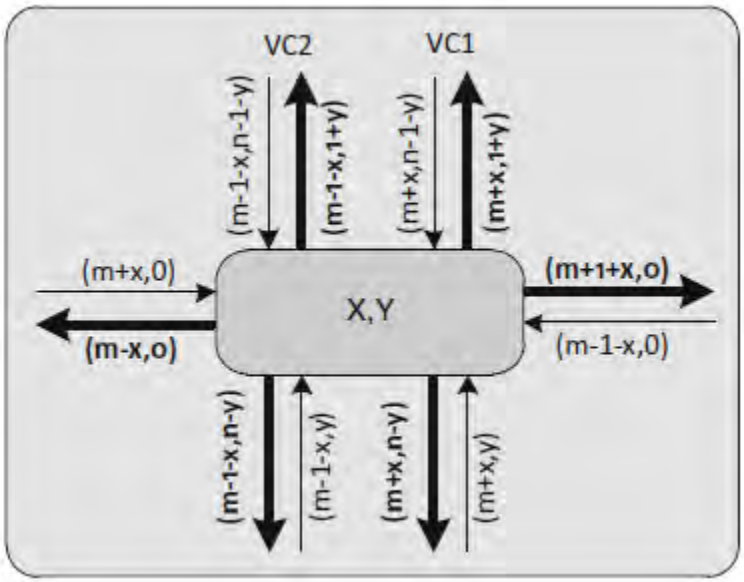
Χρησιμοποιούμε ένα και δύο εικονικά κανάλια κατά μήκος των διαστάσεων X και Y, αντίστοιχα, στα οποία μπορεί να σχηματιστούν τέσσερις κύκλοι στο δίκτυο εικόνα 2 α). Για να αποφευχθεί το αδιέξοδο, απαγορεύεται μία στροφή από κάθε κύκλο που φαίνεται στην εικόνα 4.2.β. Οι απαγορευμένες στροφές σε κάθε εικονικό κανάλι λαμβάνονται από τη μέθοδο Mad-y [13]. Με βάση αυτό το μοντέλο σειράς, επιτρέπονται οι στροφές E-N1, E-S1, N1-E, N2-E,

S1-E και S2-E για πακέτα που κινούνται ανατολικά ενώ οι στροφές W-S1, W-N1, W-N2, N2-W και S2-W είναι επιτρεπτές για πακέτα που κινούνται προς τα δυτικά.



Εικόνα 4.2 α) Τέσσερις αφηρημένοι κύκλοι όταν χρησιμοποιούνται ένα και δύο εικονικά κανάλια κατά μήκος των διαστάσεων X και Y. β) επιτρεπόμενες και απαγορευμένες στροφές αλγορίθμου RR-2D, παρόμοιες με αυτές του Mad-y [10]

Για να αποδείξουμε ότι ο αλγόριθμός μας αποφεύγει τη δημιουργία αδιεξόδου, χρησιμοποιούμε έναν μηχανισμό αρίθμησης παρόμοιο με τη μέθοδο Mad-y [13]. Αυτός ο μηχανισμός αρίθμησης δείχνει ότι όλες οι στροφές πραγματοποιούνται μόνο κατά αύξουσα σειρά και έτσι δεν μπορεί να σχηματιστεί κύκλος στο δίκτυο. Ένας διψήφιος αριθμός (a, b) αντιστοιχεί σε κάθε κανάλι εξόδου ενός δρομολογητή σε ένα δίκτυο n×m mesh. Σύμφωνα με τον μηχανισμό αρίθμησης, μια στροφή που συνδέει το κανάλι εισόδου (Iα, Ib) στο κανάλι εξόδου (Oa, Ob) ονομάζεται αύξουσα στροφή όταν (Oa> Ia) ή (Oa = Ia). Η εικόνα 4.3 δείχνει την αρίθμηση των καναλιών ενός δρομολογητή στη θέση (x, y).



Εικόνα 4.3 Ο μηχανισμός αρίθμησης της RR-2D

Χρησιμοποιώντας αυτόν τον μηχανισμό αρίθμησης, διασφαλίζεται ότι όλες οι επιτρεπόμενες στροφές λαμβάνονται αυστηρά με την επιτρεπόμενη αύξουσα σειρά, έτσι ώστε ο αλγόριθμος δρομολόγησης να μην καταλήγει σε αδιέξοδο. Για παράδειγμα, εάν ληφθεί υπόψη η σειρά E-N1 (δηλαδή ένα πακέτο που κινείται από την ανατολική κατεύθυνση προς τη βόρεια κατεύθυνση χρησιμοποιώντας το πρώτο εικονικό κανάλι), το δυτικό κανάλι εισόδου με την ετικέτα (Ia=m+x, Ib=0) συνδέεται με το πρώτο εικονικό κανάλι της βόρειας θύρας εξόδου που έχει την ετικέτα (Oa =m+x, Ob=1+y). Αυτή η στροφή λαμβάνει χώρα με αύξουσα σειρά από ((Oa = Ia) και (Ob> Ib)). Ομοίως, όλες οι άλλες επιτρεπόμενες στροφές λαμβάνονται κατά την αύξουσα σειρά.

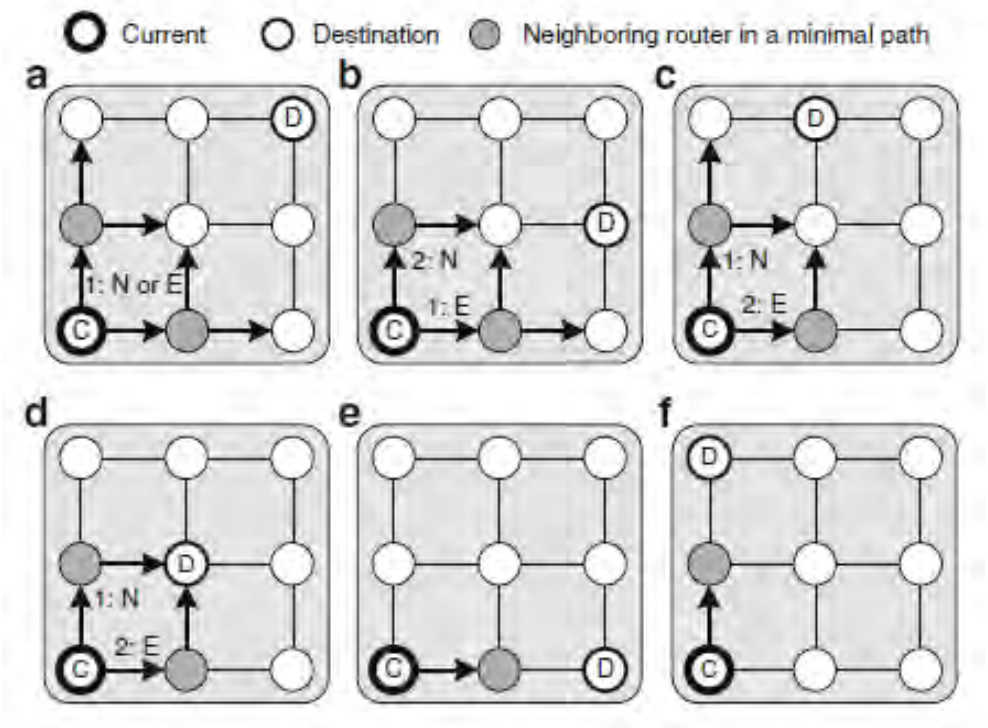
4.2.2 Αξιόπιστη δρομολόγηση σε δίκτυο 2D Mesh

Οι αλγόριθμοι δρομολόγησης ανεκτικοί σε σφάλματα είναι συνήθως πολύ περίπλοκοι. Υπάρχουν όμως και κάποιοι αλγόριθμοι που είναι πολύ απλοί και μπορούν εύκολα να εφαρμοστούν. Η βασική ιδέα πίσω από αυτούς τους

αλγόριθμους είναι η διατήρηση της προσαρμοστικότητας των πακέτων όσο το δυνατόν περισσότερο. Αυτό σημαίνει ότι για δύο ελάχιστες κατευθύνσεις, για την παράδοση ενός πακέτου, θα πρέπει κατά προτίμηση να επιλέγεται μία κατεύθυνση στην οποία το πακέτο έχει ακόμα μερικές εναλλακτικές διαδρομές για να φτάσει στον δρομολογητή προορισμού. Αυτή η απλή ιδέα αποφεύγει τη λήψη περιττών μη-ελάχιστων διαδρομών και μειώνει την πολυπλοκότητα των αλγορίθμων που είναι ανεκτικοί σε σφάλματα.

Ας εξετάσουμε τα παραδείγματα της εικόνας 4.4 όπου ένα πακέτο στέλνετε από τον τρέχοντα δρομολογητή C στον δρομολογητή προορισμού D. Στην εικόνα 4.4.α, το πακέτο μπορεί να αποσταλεί μέσω δύο ελάχιστων διευθύνσεων (E και N) για να φτάσει στον προορισμό Δρομολογητή Δ. Με την αποστολή του πακέτου σε οποιαδήποτε από τις κατευθύνσεις, το πακέτο θα έχει δύο ελάχιστες διαδρομές για να φτάσει στον δρομολογητή προορισμού. Επομένως, αν μια διαδρομή έχει ελαττωματικό κόμβο, υπάρχει η δυνατότητα να στείλουμε ένα πακέτο μέσω της άλλης διαδρομής. Ωστόσο, στην Εικόνα 4.4.β, εάν το πακέτο αποστέλλεται στον βόρειο γειτονικό δρομολογητή, θα έχει μόνο μία επιλογή να φτάσει στον δρομολογητή προορισμού. Επομένως, είναι καλύτερα να στείλετε το πακέτο προς την ανατολική κατεύθυνση από όπου υπάρχουν δύο ελάχιστες διαδρομές προς τον προορισμό. Στην εικόνα 4.4.γ, υπάρχουν ένα και δύο ελάχιστα μονοπάτια από τον ανατολικό και βόρειο γειτονικό δρομολογητή, αντίστοιχα, για να φτάσουν στον δρομολογητή προορισμού. Έτσι, η βόρεια κατεύθυνση είναι μια καλύτερη επιλογή για την παράδοση του πακέτου. Στην εικόνα 4.4.δ, υπάρχει μόνο μία πιθανή επιλογή για να φτάσουμε στον δρομολογητή προορισμού είτε από τον ανατολικό είτε βόρειο γειτονικό δρομολογητή. Στην εικόνα 4.4.ε, στ, υπάρχει μόνο μια ελάχιστη διαδρομή για να φτάσει στον προορισμό και το πακέτο πρέπει να δρομολογηθεί μέσω αυτής της μοναδικής διαδρομής. Σε αυτήν την περίπτωση, αν υπάρχει σφάλμα στη διαδρομή, το πακέτο πρέπει να λάβει μια μη ελάχιστη διαδρομή για να παρακάμψει το σφάλμα. Στην

πραγματικότητα, παίρνοντας μια διαδρομή δρομολόγησης παρόμοια με την εικόνα 4.4.β, γ, τα πακέτα δεν χάνουν την προσαρμοστικότητά τους και έτσι δεν θα αντιμετωπίσουν παρόμοιες καταστάσεις όπως στην εικόνα 4.4.ε, στ.



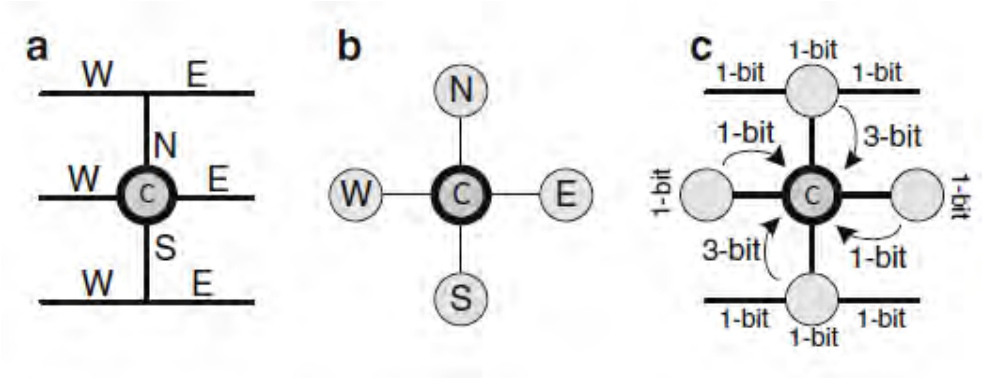
Εικόνα 4.4 Η βασική ιδέα της μεθόδου RR-2D

Γενικά, τα πακέτα δρομολογούνται μέσα στο δίκτυο χρησιμοποιώντας τις επιτρεπόμενες στροφές που προσφέρει ένας πλήρως προσαρμοστικός αλγόριθμος δρομολόγησης. Η προσαρμοστικότητα περιορίζεται όταν η απόσταση μεταξύ του δρομολογητή προέλευσης και του δρομολογητή προορισμού είναι ένα σε μία τουλάχιστον διάσταση. Το RR-2D αποφεύγει στο να μειωθεί η απόστασης σε μηδέν σε μία κατεύθυνση όταν η απόσταση στην άλλη κατεύθυνση είναι μεγαλύτερη από ένα.

4.2.3 Τεχνική παρακολούθησης και διαχείρισης σφαλμάτων

Υποθέτουμε ότι τα σφάλματα ανιχνεύονται από μια τεχνική ανίχνευσης σφαλμάτων και αποθηκεύουμε αυτές τις πληροφορίες σε κάθε δρομολογητή. Η μονάδα δρομολόγησης θα πρέπει επίσης να δέχεται όλες τις πληροφορίες σφάλματος δρομολογητών και συνδέσμων.

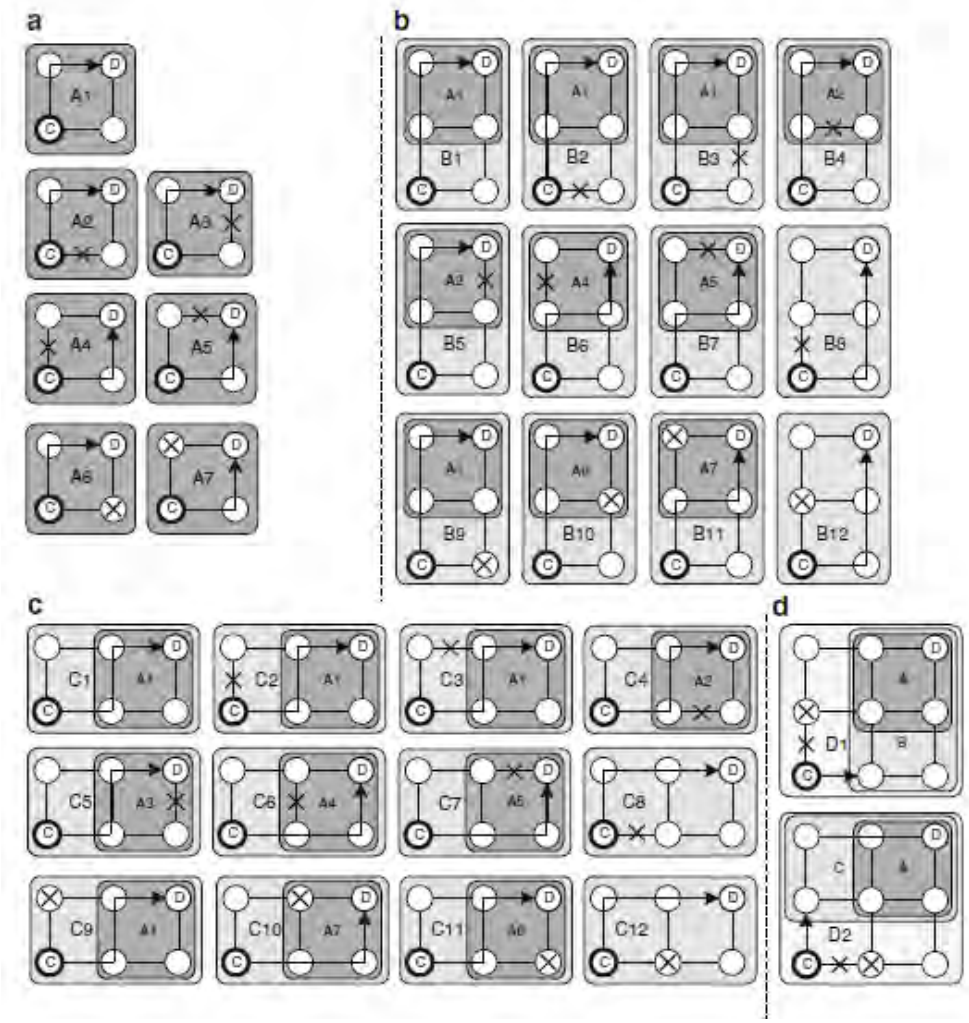
Στον αλγόριθμο RR-2D κάθε κόμβος θα πρέπει να γνωρίζει το πολύ τις καταστάσεις των οκτώ γύρω συνδέσμων για να μπορεί να δρομολογήσει τα δεδομένα (εικόνα 4.5.α). Επιπλέον, για να παρακάμψει ένα ελαττωματικό δρομολογητή, αρκεί να ενημερώνεται ο κάθε δρομολογητής για την κατάσταση των 4 γειτονικών δρομολογητών (εικόνα 4.5.β). Όλες οι πληροφορίες που συλλέγονται για κάθε κόμβο από τον RR-2D φαίνονται στην εικόνα 4.5.γ. Αυτές οι πληροφορίες θα πρέπει να μεταφερθούν στον συγκεκριμένο δρομολογητή χρησιμοποιώντας έναν μηχανισμό παρακολούθησης και διαχείρισης. Κάθε δρομολογητής έχει επίγνωση του σφάλματος στους γειτονικούς συνδέσμους του, αλλά θα πρέπει να ενημερώνεται για τυχόν σφάλματα και από τους άλλους συνδέσμους και δρομολογητές. Όπως φαίνεται στην εικόνα 4.5.γ, ένα 3-bit καλώδιο χρησιμοποιείται για τη μεταφορά των πληροφοριών σφάλματος του βορεινού γειτονικού δρομολογητή και των ανατολικών και δυτικών συνδέσεών του με τον τρέχοντα δρομολογητή. Παρομοίως, χρησιμοποιείται 3-bit για τη μεταφορά των πληροφοριών από τη νότια κατεύθυνση. Το καλώδιο 1-bit είναι αρκετό για να μεταφέρει τις πληροφορίες των ανατολικών και δυτικών γειτονικών δρομολογητών στον τρέχοντα δρομολογητή.



Εικόνα 4.5 Καταστάσεις 8 links και 4 δρομολογητών που απαιτεί το RR-2D

4.2.4 Ανεκτικότητα ελαττωματικών συνδέσμων και δρομολόγηση στον RR-2D

Ο απλός κανόνας είναι να ελέγξει ο αλγόριθμος RR-2D αν υπάρχει δυνατότητα αποστολής του πακέτου μέσω της διάστασης με τη μεγαλύτερη απόσταση όταν ο δρομολογητής προορισμού απέχει ένα hop σε μία διάσταση. Το πακέτο αποστέλλεται στη διάσταση με μεγαλύτερη απόσταση αν ο άμεσος σύνδεσμος του δρομολογητή κατά την κατεύθυνση αυτή είναι ελαττωματικός. Σε όλες τις άλλες περιπτώσεις, τα πακέτα δεν έχουν περιορισμό δρομολόγησης. Οι διαφορετικές θέσεις ελαττωματικών συνδέσμων και δρομολογητών απεικονίζονται στην εικόνα 4.6.



Εικόνα 4.6 Tolerating faulty routers and links by RR-2D

Όπως φαίνεται στην εικόνα 4.6, όταν η απόσταση κατά μήκος των δύο διαστάσεων X και Y φτάνει στο ένα (X-dir = 1 και Y-dir = 1), θα υπάρχουν έξι διαφορετικές θέσεις ελαττωματικών συνδέσμων (δύο δρομολογητές και τέσσερις κόμβοι). Σύμφωνα με τον RR-2D, η ανατολική και η βόρεια κατεύθυνση έχουν την ίδια προτεραιότητα καθώς δεν υπάρχουν εναλλακτικές επιλογές. Πάντως, σύμφωνα με το μηχανισμό κατανομής βλάβης (εικόνα 4.5.γ), ο τρέχων δρομολογητής γνωρίζει σχετικά με το ελαττωματικό σύνδεσμο αριστερά στη διαδρομή BA και ως εκ τούτου το γνωρίζει και ο γειτονικός δρομολογητής N, παρόλο που σε γενικές γραμμές δεν γνωρίζει

σχετικά με την κατάσταση βλάβης σε όλη το μονοπάτι ΕΝ. Επομένως, η διαθεσιμότητα της διαδρομής ΝΕ και του Ν δρομολογητή ελέγχονται και αν δεν περιέχουν ελαττωματικό σύνδεσμο το πακέτο αποστέλλεται προς τη βόρινή κατεύθυνση, αλλιώς το πακέτο αποστέλλεται προς την κατεύθυνση ανατολική. Στα κομμάτια Α1, Α2, Α3, Α6 και της εικόνας 6 α), η διαδρομή ΝΕ και ο Ν δρομολογητής δεν έχουν ελαττωματικούς συνδέσμους και το πακέτο στέλνεται προς τα βόρεια κατεύθυνση. Στα κομμάτια Α4, Α5, Α7 της ίδιας εικόνας, το πακέτο παραδίδεται στον ανατολική κατεύθυνση επειδή είτε η διαδρομή ΝΕ είτε ο Ν δρομολογητής είναι ελαττωματικός.

Στην εικόνα 4.6.β, όταν το X-dir = 1 και Y-dir = 2, ελαττωματικό θα μπορούσε να είναι μια από τις επτά διαφορετικές θέσεις κόμβων και τέσσερις διαφορετικοί δρομολογητές. Σε όλες τις περιπτώσεις, οι διαθεσιμότητες του συνδέσμου Ν και του Ν δρομολογητή εξετάζονται πριν από αυτών που βρίσκονται στην ανατολή. Στα κομμάτια Β1, Β2, Β3, Β4, Β5, Β6, Β7, Β9, Β10, και Β11 της εικόνας 4.6.β, το πακέτο στέλνεται προς τα βόρεια κατεύθυνση καθώς τόσο ο σύνδεσμος όσο και ο δρομολογητής είναι μη-ελαττωματικοί. Στα κομμάτια Β8 και Β12 της εικόνας 4.6.β, το πακέτο πρέπει να κατευθυνθεί προς την ανατολική κατεύθυνση για να παραδοθεί στον δρομολογητή προορισμού. Στην εικόνα 4.6.γ, όταν το X-dir = 2 και Y-dir = 1, η διαθεσιμότητα της σύνδεσης Ε και ο Ε δρομολογητής θα πρέπει να ελεγχθούν πριν από τη σύνδεση Ν και τον Ν δρομολογητή. Ως εκ τούτου, στα κομμάτια C1, C2, C3, C4, C5, C6, C7, C9, C10, και C11 της εικόνας 4.6.γ, το πακέτο στέλνεται προς την ανατολική κατεύθυνση αφού τόσο η Ε σύνδεσμος όσο και ο Ε δρομολογητής είναι μη ελαττωματικοί. Στην κομμάτια C8 και C12, το πακέτο παραδίδεται προς τα βορινή κατεύθυνση και έτσι το σφάλμα παρακάμπτεται. Σε όλες τις άλλες περιπτώσεις (όταν το X-dir>=2 και Y-dir>=2 της εικόνας 4.6.δ, το πακέτο στέλνεται στην μη ελαττωματική κατεύθυνση. Στο επόμενο hop, οι περιπτώσεις είναι παρόμοιες με αυτές της εικόνας 4.6.β, γ. Η ίδια διαδικασία μπορεί να εφαρμοστεί στις περιπτώσεις

northwest-, southeast-, και southwest-Ward. Η εικόνα 4.7 δείχνει τον ψευδοκώδικα του RR-2D αλγόριθμο δρομολόγησης.

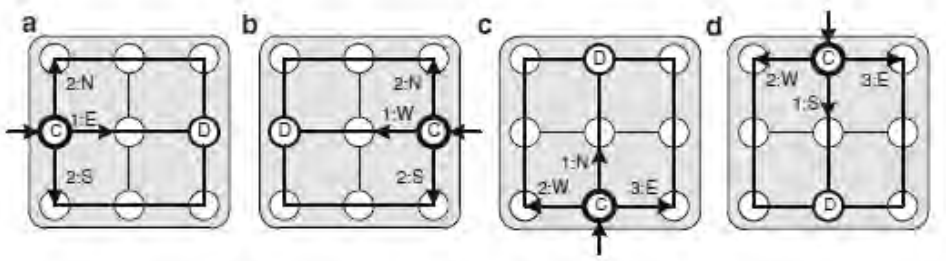
```
Definitions: Xc,Yc,Xd,Yd: X and Y coordinates of current and destination
routers
                    **_____**
position <= {NE,NW,SE,SW} according to the source and destination pos.

x_dir <= E when Xd > Xc else W;
y_dir <= N when Yd > Yc else S;
delta_x <= Xd - Xc when Xd>Xc else Xc-Xd;
delta_y <= Yd - Yc when Yd>Yc else Yc-Yd;
vc <= vc1 when position={E,NE,SE} else      --eastward packets
      vc2 when position={W,N,S,NW,SW};     --westward packets

if position={NE, NW, SE, or SW} then
   if (delta_x>=1 and delta_y=0) then select <= x_dir;
   elsif (delta_x=0 and delta_y>=1) then select <= y_dir(vc);
   elsif (delta_x=1 and delta_y>=1) then
         if neighbor(y_dir)=faulty or link(y_dir)=faulty then
                  select <= x_dir;
         else select <= y_dir(vc); end if;
   elsif (delta_x>1 and delta_y=1) then
         if neighbor(x_dir)=faulty or link(x_dir)=faulty then
                  select <= y_dir(vc);
         else select <= x_dir; end if;
   else
         if neighbor(x_dir)=faulty or link(x_dir)=faulty then
                  select <= y_dir(vc);
         elsif neighbor(y_dir)=faulty or link(x_dir)=faulty then
                  select <= x_dir;
         else select <= x_dir or y_dir(vc); end if;
   end if;
end if;
```

Εικόνα 4.7 Ψευδοκώδικας του αλγορίθμου RR-2D για δρομολογήσεις northeast, northwest, southeast και southwest

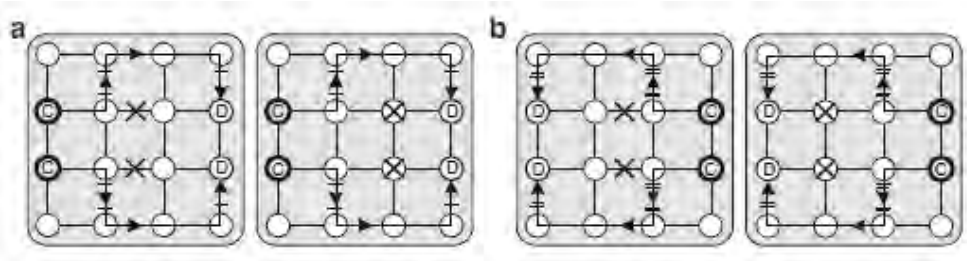
Όταν ένα πακέτο πρέπει να προωθηθεί στην κατεύθυνση είναι Ανατολή-, Δύση-, Βορράς- ή Νότια- και υπάρχει μια ελαττωματική σύνδεση ή δρομολογητής στο μονοπάτι, το πακέτο θα πρέπει να αλλάξει το δρομολόγιο μέσω μιας μη ελάχιστης διαδρομής. Όπως φαίνεται στην εικόνα 4.8.α, για την προς ανατολή διαδρομή του πακέτου, ελέγχεται η πρώτη η σύνδεση και ο δρομολογητής στα ανατολικά και εάν είναι μη-ελαττωματικοί, το πακέτο στέλνεται προς αυτή την κατεύθυνση. Ωστόσο, αν είτε η σύνδεση ή είτε ο δρομολογητής είναι ελαττωματικός, το πακέτο παραδίδεται προς τα βόρεια ή νότια κατεύθυνση ανάλογα με τη συμφόρηση των πακέτων στο δίκτυο. Τα πακέτα με προορισμό δυτικό δρομολογητή έχουν την ίδια ακριβώς συμπεριφορά (εικόνα 4.8.β). Εάν ένα πακέτο με κατεύθυνση το βορρά αντιμετωπίζει ένα σφάλμα στη βόρεια σύνδεση ή δρομολογητή (εικόνα 4.8.γ), η δυτική κατεύθυνση ελέγχεται νωρίτερα από την ανατολική κατεύθυνση. Είναι bedeutet, dass αλλαγή δρομολογίου μέσα από την ανατολική κατεύθυνση γίνεται μόνο όταν η βλάβη βρίσκεται στο αριστερό άκρο. Το ίδιο εφαρμόζεται και για τα πακέτα με κατεύθυνση το νότο (εικόνα 4.8.δ).



Εικόνα 4.8 Παράκαμψη σφαλμάτων όταν ο προορισμός βρίσκεται στο α) ανατολικά, β) δυτικά, γ) βόρεια, δ) νότια

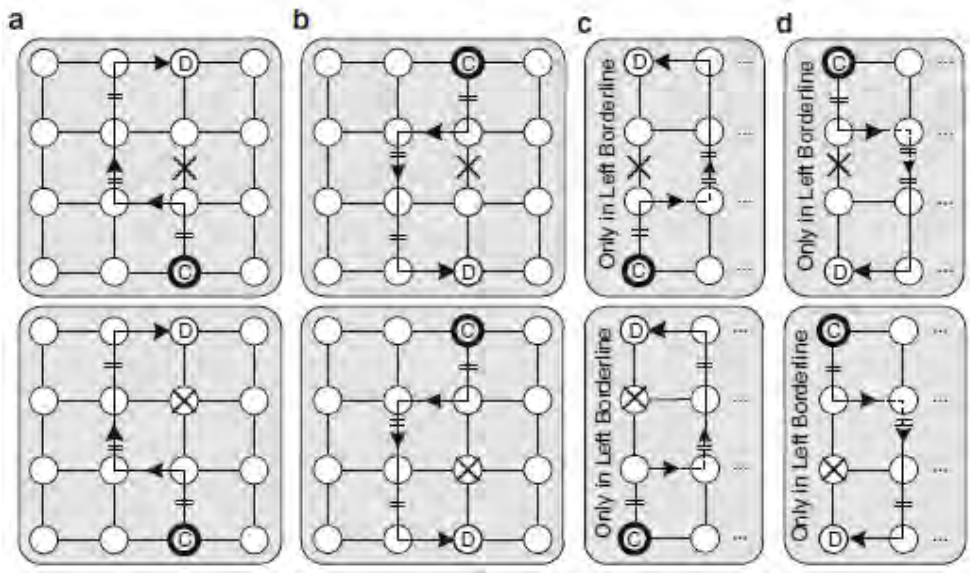
Στη συνέχεια, πρέπει να εξετάσουμε αν όλες οι απαιτούμενες στροφές για την παράκαμψη των σφαλμάτων είναι στο σύνολο των επιτρεπόμενων στροφών. Κατά τη διερεύνηση των απαιτούμενων στροφών, παρατηρούμε τα πακέτα που κινούνται ανατολικά χρησιμοποιούν τις στροφές E-N1, N1-E, E-S1 και S1-E για να παρακάμψουν ένα ελαττωματικό σύνδεσμο ή δρομολογητή

(εικόνα 4.9.α). Όλες αυτές οι στροφές είναι στο σύνολο των επιτρεπόμενων στροφών. Ομοίως, όπως φαίνεται στην εικόνα 4.9.β, όλες οι απαιτούμενες στροφές από τα πακέτα που βρίσκονται δυτικά επιτρέπεται (δηλ. W-N2, N2-W, W S2, και S2-W).



Εικόνα 4.9 Ανοχή σφάλματος σε πακέτα που κινούνται α) προς την ανατολή β) προς τη δύση

Θα πρέπει επίσης να αποδειχθεί ότι τα πακέτα που δρομολογούνται στο δίκτυο από το βορρά ή το νότο δρομολογούνται χωρίς να δημιουργεί αδιέξοδο. Όπως απεικονίζεται στην εικόνα 4.10.α, και β, χρησιμοποιούν τις επιτρεπόμενες στροφές N2-W, W-N2, N2-E, S2-W, W S2, S2-E για να παρακάμψουν τα σφάλματα. Ωστόσο, όταν οι δρομολογητές πηγής και προορισμού που βρίσκεται στο αριστερό όριο του NoC υπάρχει μια ελαττωματική σύνδεση ή ελαττωματικός δρομολογητής στη διαδρομή, οι απαιτούμενες στροφές είναι οι N2-E, E-N2, N2-W, S2-E, E-S2, και S2-W. Μεταξύ αυτών, οι στροφές E-N2 και E-S2 είναι μη επιτρεπόμενες από το σύστημα, αλλά όπως υποδεικνύεται [19] πλήρης κύκλος δεν μπορεί να σχηματιστεί στις οριακές περιπτώσεις και οι μη επιτρεπόμενες στροφές μπορούν να ληφθούν με ασφάλεια.



Εικόνα 4.10 Ανοχή ενός σφάλματος σε πακέτα που κινούνται από α) τον βορρά β) το νότο και γ), δ) πακέτα που κινούνται στα όρια του NoC

Ο υπολειπόμενος αλγόριθμος δρομολόγησης RR-2D να την αποφυγή ελαττωματικών συνδέσεων και δρομολογητών για τα πακέτα east-, west-, north-, και south- φαίνεται στην εικόνα 4.11 (συνέχεια της εικόνας 4.7).

```
if position={E or W} then
    if delta_y=0 then
        if neighbor(x_dir)=faulty or link(x_dir)=faulty then
            select <= N(vc) or S(vc) based on their availability;
        else select <= x_dir; end if;
    else
        if neighbor(y_dir)=dest or link(y_dir)=faulty then
            select <= y_dir(vc);
        else select <= x_dir; end if;
     end if;
elsif position={N or S} then
    if delta_x=0 then
        if neighbor(y_dir)=faulty or link(y_dir)=faulty then
            if Xc/=0 then select <= west; else select <= east; end if;
        else select <= y_dir(vc); end if;
    else
        if inPort/= {E,W} and
          (neighbor(x_dir)=non-faulty or link(x_dir)=faulty) then
            select <= x_dir;
        else select <= y_dir(vc); end if;
    end if;
end if;
```

Εικόνα 4.11 RR-2D για πακέτα east-, west-, north-, και south-

4.3 Αλγόριθμος δρομολόγησης Fault-Tolerant σε 3D Mesh δίκτυο

Αρχικά, παρουσιάζουμε έναν πλήρως προσαρμοστικό αλγόριθμο δρομολόγησης σε ένα δίκτυο 3D mesh χρησιμοποιώντας δύο, δύο και τέσσερα εικονικά κανάλια. Στη συνέχεια, θα δείξουμε πώς τα πακέτα μπορούν να αλλάξουν μεταξύ των εικονικών καναλιών για να επιτύχουν μια καλύτερη ανεκτικότητα απέναντι στα σφάλματα και θα εξηγούμε την τεχνική παρακολούθησης και διαχείρισης βλαβών.

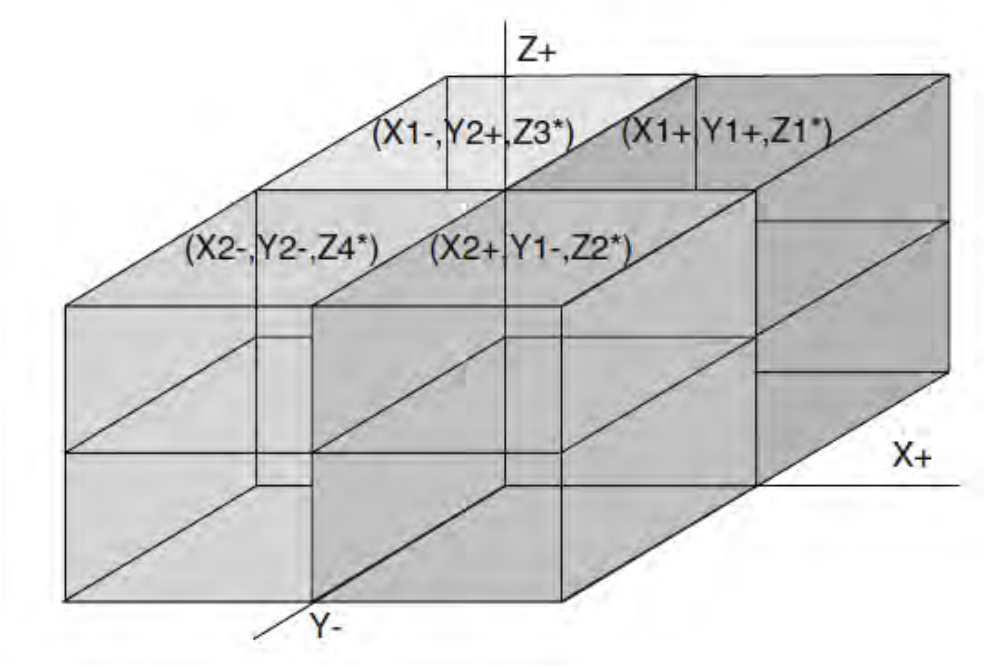
4.3.1 Πλήρως προσαρμοστική δρομολόγηση σε δίκτυο 3D Mesh

Ένα δίκτυο 3D μπορεί να χωριστεί σε οκτώ υποδικτυακά δίκτυα: ((X+)(Y+)(Z+), (X+)(Y+)(Z-), (X+)(Y−)(Z+), (X+)(Y−)(Z−), (X−)(Y+)(Z+), (X−)(Y+)(Z−), (X−)(Y-)(Z+), και (X−)(Y−)(Z−)), όπου "+", "-" αντιπροσωπεύουν τα κανάλια κατά μήκος των θετικών και αρνητικών διευθύνσεων, αντίστοιχα. Ένας απλός τρόπος για την εφαρμογή ενός πλήρως προσαρμοστικού αλγορίθμου δρομολόγησης είναι η εκχώρηση ενός ξεχωριστού συνόλου εικονικών καναλιών σε κάθε υποδίκτυο όπως: ((X1+)(Y1+)(Z1+), (X2+)(Y2+)(Z1−), (X3+)(Y1−)(Z2+), (X4+)(Y2−)(Z2−), (X1−)(Y3+)(Z3+), (X2−)(Y4+)(Z3−), (X3−)(Y3−)(Z4+), και (X4−)(Y4−)(Z4−)) όπου οι αριθμοί υποδεικνύουν τον αριθμό εικονικού καναλιού κατά μήκος κάθε διάστασης. Με αυτήν την αντιστοίχιση καναλιού, χρειάζονται τέσσερα εικονικά κανάλια για κάθε διάστασης και για το δίκτυο.

Στον αλγόριθμο DyXYZ [14], ο αριθμός των εικονικών καναλιών μειώνεται σε δύο κατά μήκος κάθε διάστασης (έτσι απαιτούνται τέσσερα, τέσσερα και δύο εικονικά κανάλια κατά μήκος των διαστάσεων X, Y και Z). Στη μέθοδο αυτή, ολόκληρο το δίκτυο χωρίζεται περαιτέρω σε δύο κύρια υποδίκτυα, το καθένα από τα οποία έχει τέσσερα υποδίκτυα. Ο διαχωρισμός του δικτύου σε δύο μέρη μπορεί να γίνει κατά μήκος μιας διάστασης, μειώνοντας τον αριθμό των εικονικών καναλιών από τέσσερα σε δύο για αυτήν τη διάσταση. Δεδομένου ότι δύο βασικά υποδίκτυα είναι χωρισμένα, το δίκτυο παραμένει ελεύθερο αδιεξόδου.

Ο αριθμός των εικονικών καναλιών μπορεί να μειωθεί περαιτέρω σε δύο, δύο και τέσσερα εικονικά κανάλια κατά μήκος των διαστάσεων X, Y και Z (δηλ. Δύο εικονικά κανάλια μικρότερα από το DyXYZ). Σε αυτή τη μέθοδο, το δίκτυο χωρίζεται σε τέσσερα υποδίκτυα (εικόνα 4.12) ως: ((X+)(Y+)(Z*), (X−)(Y+)(Z*), (X+)(Y−)(Z*), και (X−)(Y−)(Z*)), Όπου τα "+", "-"

αντιπροσωπεύουν τα κανάλια κατά μήκος των θετικών και αρνητικών διευθύνσεων, αντίστοιχα, και το "*" σημαίνει θετικές και αρνητικές διευθύνσεις (Δηλαδή ένα αμφίδρομο κανάλι). Η απουσία αδιεξόδου μπορεί να πραγματοποιηθεί αφού αποδειχθεί ότι τα υποδικτυα είναι διαφορετικά μεταξύ τους και κανένας κύκλος δεν μπορούν να διαμορφωθεί μέσα σε κάθε υποδίκτυο. Η εκχώρηση εικονικού καναλιού αυτού του αλγορίθμου είναι ως ακολούθως: ((X1+)(Y1+)(Z1*), (X2+)(Y1−)(Z2*), (X1−)(Y2+)(Z3*), και (X2−)(Y2−)(Z4*)), όπου οι αριθμοί υποδεικνύουν τον αριθμό εικονικού καναλιού που αντιστοιχεί σε μια διάσταση.



Εικόνα 4.11 Δημιουργία τεσσάρων διαφορετικών υποδικτύων

Θεώρημα 1 Αποφυγή αδιεξόδου σε κάθε υποδίκτυο

Για να διαμορφωθεί ένας κύκλος θα πρέπει τα πακέτα να είναι ικανά να παίρνουν τόσο θετικές όσο και αρνητικές κατευθύνσεις κατά μήκος τουλάχιστον δύο διαστάσεων. Για παράδειγμα, για να σχηματίσουμε έναν κύκλο σε ένα επίπεδο XY, είναι απαραίτητο να ακολουθήσουμε τις

κατευθύνσεις X+, X-, Y+, και Y-. Ομοίως, για να σχηματίσουμε έναν κύκλο σε ένα επίπεδο YZ, θα πρέπει να υπάρχει η δυνατότητα λήψης οδηγιών Y+, Y-, Z+, και Z-. Τέλος, για να σχηματίσουμε έναν κύκλο σε ένα επίπεδο XZ, οι κατευθύνσεις X+, X-, Z+ και Z- θα πρέπει να ληφθούν με πακέτα. Όπως μπορεί να εξασφαλιστεί από τέσσερα διαφορετικά υποδικτυακά δίκτυα, μόνο ένα ζεύγος κατά μήκος της διάστασης Z (Z*: Z-, Z +) πραγματοποιείται σε καθένα από τα τέσσερα υποδίκτυα και επομένως εξασφαλίζεται η απουσία αδιεξόδου.

Θεώρημα 2 Απουσία αδιεξόδου μεταξύ των υποδικτύων

Για να αποδειχθεί ότι το δίκτυο είναι ελεύθερο αδιεξόδου μεταξύ των υποδικτύων, αρκεί να δείξουμε τα ότι διαφορετικά υποδίκτυα είναι διαχωρισμένα μεταξύ τους κατά μήκος κάθε διάστασης. Με τη σύγκριση ανά ζεύγη μεταξύ των δύο υποδικτύων, μπορεί εύκολα να αποδειχθεί ότι είτε η κατεύθυνση είτε ο αριθμός εικονικού καναλιού διαφέρει κατά μήκος κάθε διάστασης. Για παράδειγμα, το υποδίκτυο 1 (X1+)(Y1+)(Z1 *) και το υποδίκτυο 3 (X1-)(Y2 +)(Z3*) διαχωρίζονται κατά μήκος της διάστασης X, δεδομένου ότι το υποδίκτυο 1 καλύπτει τη θετική κατεύθυνση του εικονικού καναλιού 1 (X1+) και το υποδίκτυο 3 καλύπτει την αρνητική κατεύθυνση του εικονικού καναλιού 1 (X1-). Τα δύο υποδίκτυα είναι επίσης διαχωρισμένα κατά μήκος της διάστασης Y από τη στιγμή που το εικονικό κανάλι 1 χρησιμοποιείται στο υποδίκτυο 1 (Y1+) και το εικονικό κανάλι 2 χρησιμοποιείται στο υποδίκτυο 3 (Y2+). Ομοίως, δύο υποδίκτυα χρησιμοποιούν διαφορετικό αριθμό εικονικού καναλιού κατά μήκος της διάστασης Z (Z1* και Z3*). Αυτή η σύγκριση μπορεί να γίνει για οποιαδήποτε άλλα ζεύγη υποδικτύων (Πίνακας 4.1).

Subnetwork A	Subnetwork B	X dimension	Y dimension	Z dimension
(X1+)(Y1+)(Z1*)	(X1−)(Y2+)(Z3*)	Differs in vc direction	Differs in vc number	Differs in vc number
(X1+)(Y1+)(Z1*)	(X2+)(Y1−)(Z2*)	Differs in vc direction	Differs in vc number	Differs in vc number
(X1+)(Y1+)(Z1*)	(X2−)(Y2−)(Z4*)	Differs in vc direction	Differs in vc number	Differs in vc number
(X1−)(Y2+)(Z3*)	(X2+)(Y1−)(Z2*)	Differs in vc direction	Differs in vc number	Differs in vc number
(X1−)(Y2+)(Z3*)	(X2−)(Y2−)(Z4*)	Differs in vc direction	Differs in vc number	Differs in vc number
(X2+)(Y1−)(Z2*)	(X2−)(Y2−)(Z4*)	Differs in vc direction	Differs in vc number	Differs in vc number

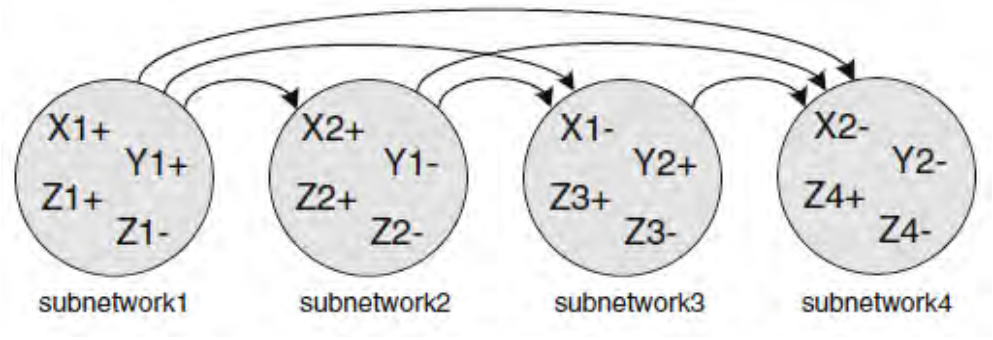
Πίνακας 4.1 Αντιστοίχιση εικονικών καναλιών και υποδικτύων

Λαμβάνοντας υπόψη το Θεώρημα 1 και το Θεώρημα 2, αποδεικνύουμε ότι το δίκτυο είναι ελεύθερο αδιεξόδων για κάθε υποδίκτυο και από την άλλη πλευρά, τα διαφορετικά υποδίκτυα είναι διαχωρισμένα μεταξύ τους, επομένως το όλο δίκτυο είναι ελεύθερο αδιεξόδου.

Αποδείξαμε ότι τα τέσσερα υποδίκτυα είναι διαφορετικά μεταξύ τους. Βάσει αυτής της απόδειξης, όλα τα πακέτα ανήκουν σε ένα υποδίκτυο και δεν μπορούν να μεταβούν στα άλλα. Ωστόσο, περιορίζει την ικανότητα αντοχής του συστήματος σε σφάλματα, καθώς τα πακέτα δεν μπορούν να χρησιμοποιήσουν τα κανάλια από τα άλλα υποδίκτυα σε περίπτωση βλαβών.

Για να βελτιώσουμε την ικανότητα αντοχής σε σφάλματα, ενεργοποιούμε την εναλλαγή πακέτων μεταξύ υποδικτύων χωρίς να δημιουργούμε κύκλους. Όπως φαίνεται στην εικόνα 4.13, τα πακέτα μπορούν να μετακινούνται μεταξύ των υποδικτύων με αυστηρά αύξουσα τάξη έτσι ώστε να μην μπορεί

να σχηματιστεί ένας κύκλος. Αυτό σημαίνει ότι τα πακέτα που μετακινούνται από το υποδίκτυο 1 μπορούν να μεταβούν σε υψηλότερα υποδικτυα (2, 3 ή 4), αλλά μετά την εναλλαγή δεν επιτρέπεται πλέον να χρησιμοποιούν κανάλια από το υποδίκτυο 1. Παρόμοια, τα πακέτα στο υποδίκτυο 2 μπορούν να μεταβούν στο υποδίκτυο 3 ή 4 χάνοντας τη δυνατότητα χρήσης των καναλιών του υποδίκτυου 1 ή του δευτερεύοντος υποδικτύου 2. Τέλος, τα πακέτα που κινούνται στο υποδίκτυο 1, το υποδίκτυο 2 ή το υποδίκτυο 3 μπορούν να μεταβούν στο υποδίκτυο 4, αλλά δεν μπορούν να χρησιμοποιήσουν κανάλια παρά μόνο από το υποδίκτυο 4. Γενικά, μετά τη χρήση ενός καναλιού σε ένα υψηλότερο υποδίκτυο, δεν επιτρέπεται η μετάβαση στο κατώτερο υποδίκτυο.



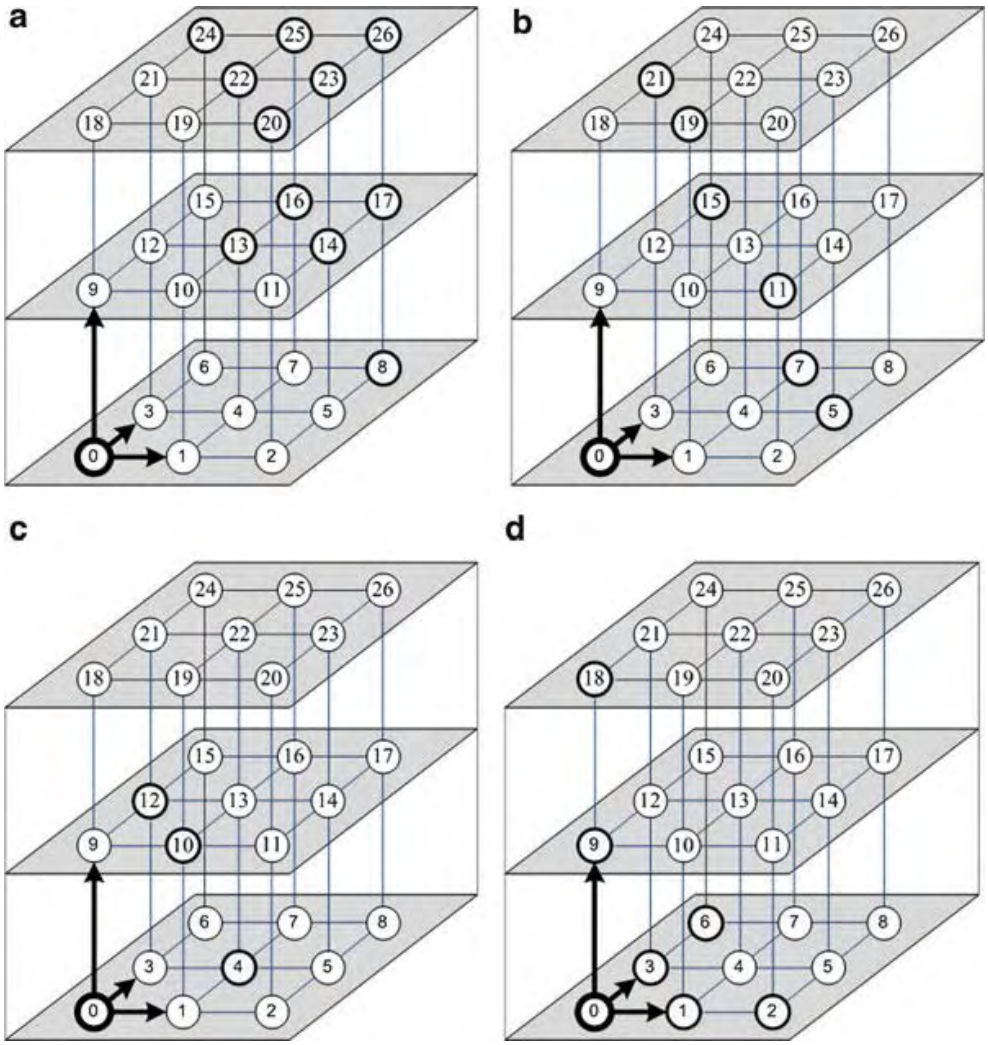
Εικόνα 4.13 Τα πακέτα μπορούν να αλλάζουν μεταξύ των υποδικτύων αλλά μόνο με την αυστηρά αύξουσα σειρά

4.3.2 Αξιόπιστη δρομολόγηση σε δίκτυο 3D Mesh

Όπως και σε ένα δίκτυο 2D mesh, προκειμένου να γίνει ανεκτικό σε σφάλματα ένα δίκτυο 3D mesh, ο πλήρως προσαρμοστικός αλγόριθμος δρομολόγησης τροποποιείται ελαφρώς. Όπως φαίνεται στην εικόνα 4.14.α, όταν μια πηγή βρίσκεται στο δρομολογητή 0 και ένας προορισμός βρίσκεται στον δρομολογητή 8, 13, 14, 16, 17, 20, 22, 23, 24, 25 ή 26, υπάρχουν τουλάχιστον δύο ελάχιστε διαδρομές για να προωθηθεί το πακέτο. Με την αποστολή ενός πακέτου σε οποιονδήποτε από τους γειτονικούς δρομολογητές που βρίσκονται στις ελάχιστες διαδρομές, θα υπάρχουν τουλάχιστον δύο

εναλλακτικές διαδρομές από τον επόμενο δρομολογητή στον προορισμό. Στο σχήμα 4.14.β, όταν ένας προορισμός βρίσκεται στον δρομολογητή 5, 7, 11, 15, 19 ή 21, υπάρχουν δύο κατευθύνσεις για την προώθηση ενός πακέτου. Ωστόσο, αποστέλλοντας το πακέτο σε μία από αυτές τις κατευθύνσεις, η προσαρμοστικότητα θα περιοριστεί σε μία διαδρομή, ενώ με την παράδοση του πακέτου προς την άλλη κατεύθυνση θα υπάρχουν τουλάχιστον δύο συντομότερες διαδρομές για να φτάσει στον προορισμό του.

Επομένως, είναι καλύτερο να επιλέξουμε μια διαδρομή που να διατηρεί την προσαρμοστικότητα των πακέτων. Στην εικόνα 4.14.γ, όταν ένας προορισμός βρίσκεται στο δρομολογητή 4, 10 ή 12, υπάρχουν και πάλι δύο επιλογές για να προωθηθεί το πακέτο. Όποια και να επιλέξουμε, η προσαρμοστικότητα θα χαθεί και έτσι δεν υπάρχει καμία προτίμηση στο πια να επιλέξουμε. Τέλος, στην εικόνα 4.14.δ, τα πακέτα δεν έχουν εναλλακτικές διαδρομές για να φτάσουν στον δρομολογητή προορισμού 1, 2, 3, 6, 9 ή 18.

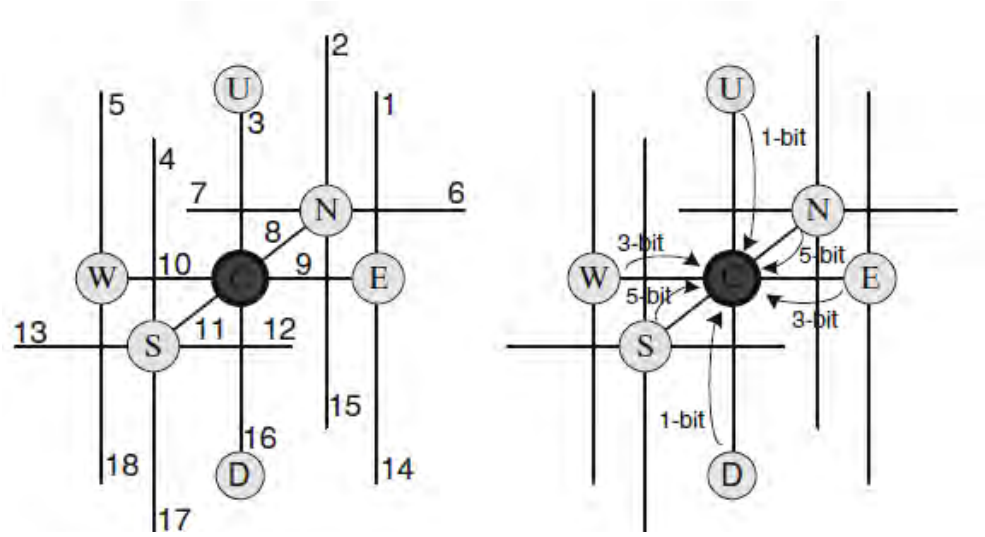


Εικόνα 4.14 Αξιόπιστη δρομολόγηση για ένα δίκτυο 3D mesh

4.3.3 Τεχνική παρακολούθησης και διαχείρισης σφαλμάτων

Για να είναι οι αλγόριθμοί μας σε βλάβες, θα πρέπει να παρέχονται ορισμένες πληροφορίες σφάλματος στη μονάδα δρομολόγησης. Υποθέτουμε ότι τα σφάλματα ανιχνεύονται από μια τεχνική ανίχνευσης σφαλμάτων και παρακολουθούμε αυτές τις πληροφορίες σε κάθε δρομολογητή. Το RR-3D πρέπει να παρακολουθεί τις ελαττωματικές καταστάσεις των έξι γειτονικών δρομολογητών του και δεκαοκτώ περιβάλλουσες συνδέσεις όπως φαίνεται στην εικόνα 4.15 σε κάθε δεδομένο δρομολογητή (δηλ. στο δρομολογητή C

όπως φαίνετε και στην εικόνα). Στους συνδέσμους περιλαμβάνονται και οι άμεσοι σύνδεσμοι που συνδέονται με κάθε γειτονικό δρομολογητή (δηλ. 3, 8, 9, 10, 11 και 16). Οι πληροφορίες αυτών των συνδέσεων παρέχονται στο δρομολογητή μέσω μιας τεχνική ανίχνευσης σφαλμάτων.



Εικόνα 4.15 Παρακολούθηση και διαχείριση σφαλμάτων στον αλγόριθμο RR-3D

Ωστόσο, οι πληροφορίες σφάλματος των άλλων συνδέσμων θα πρέπει να αποστέλλονται στον συγκεκριμένο δρομολογητή. Οι σύνδεσμοι αυτοί είναι οι εξής: τέσσερις συνδέσεις που συνδέονται με το βόρειο γειτονικό δρομολογητή (δηλαδή 2, 6, 7 και 15), τέσσερις συνδέσεις που συνδέονται με το νότιο γειτονικό δρομολογητή (δηλαδή 4, 12, 13 και 17) δύο συνδέσμους που συνδέονται με τον ανατολικό γειτονικό δρομολογητή (Δηλαδή 1 και 14) και δύο συνδέσμους που συνδέονται με το δυτικό γειτονικό δρομολογητή (δηλαδή 5 και 18). Επιπλέον, χρησιμοποιώντας μια τεχνική ανίχνευσης σφαλμάτων, ένας δρομολογητής ενημερώνεται για την κατάσταση σφάλματος του ίδιου, ενώ οι πληροφορίες σχετικά με τους γειτονικούς δρομολογητές (δηλαδή βόρεια, νότια, ανατολικά, δυτικά, επάνω και κάτω) πρέπει να μεταφερθούν στον δεδομένο δρομολογητή. Ο μηχανισμός διαχείρισης σφαλμάτων είναι υπεύθυνος για να συνδυάσει τις πληροφορίες σφάλματος σε

κάθε δρομολογητή και να τον μεταφέρει στους γειτονικούς. Σε αυτό το παράδειγμα, ένας από τους γειτονικούς δρομολογητές του Βορρά και του Νότου συνδυάζει τις πληροφορίες σφάλματος του ίδιου του δρομολογητή και των συνδεδεμένων τεσσάρων συνδέσμων και μεταφέρει μια πληροφορία 5-bit στο συγκεκριμένο δρομολογητή. Οι ανατολικοί και δυτικοί γειτονικοί δρομολογητές μεταφέρουν πληροφορία 3-bit στον δεδομένο δρομολογητή. 1-bit για την κατάσταση σφάλματος του δρομολογητή και 2-bit για τις καταστάσεις βλαβών των συνδέσμων. Από τις κατευθύνσεις προς τα επάνω και προς τα κάτω, μεταφέρονται πληροφορίες 1-bit στον δρομολογητή που δείχνει την κατάσταση σφάλματος του συνδεδεμένου δρομολογητή.

Στις πληροφορίες που πρέπει να μεταφερθούν από τον τρέχοντα δρομολογητή σε κάθε γειτονικό δρομολογητή συγκαταλέγονται:

- Η κατάσταση σφάλματος του δρομολογητή C και των τεσσάρων συνδέσεών του 3, 9, 10 και 16 μεταφέρονται στους δρομολογητές γειτονικών του Βορρά και Νότου.

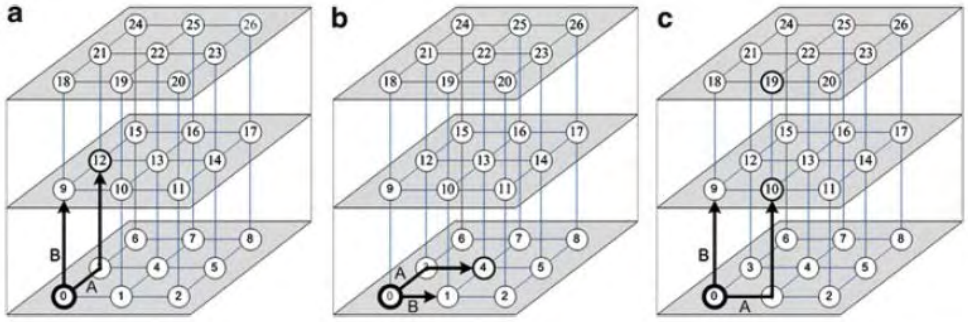
- Η κατάσταση σφάλματος του δρομολογητή C και των δύο συνδέσεών του 3 και 16 μεταφέρονται στους ανατολικούς και δυτικούς γειτονικούς δρομολογητές.

Τέλος, μόνο η κατάσταση σφάλματος του δρομολογητή C αποστέλλεται στους άνω και κάτω γειτονικούς δρομολογητές.

4.3.4 Ανίχνευση ελαττωματικών συνδέσμων και δρομολογητών από τον RR-3D

Ένα όμως από τα βασικότερα ερωτήματα παραμένει το κατά πόσο οι ελαττωματικοί σύνδεσμοι μπορούν να γίνουν ανιχνεύσιμοι με τη χρήση του RR-3D. Όταν τα πακέτα έχουν τουλάχιστον δύο ελάχιστες διαδρομές (παρόμοια με το παράδειγμα της εικόνας 4.14.α, τότε αυτές στέλλονται σε μια

κατεύθυνση με μη ελαττωματικό σύνδεσμο και δρομολογητή. Η απόφαση για τη δρομολόγηση λαμβάνεται όταν η απόσταση μεταξύ δύο κατευθύνσεων φτάσει τη μία μονάδα.



Εικόνα 4.16 RR-3D όταν η απόσταση φτάσει σε μία κατά μήκος δύο κατευθύνσεων

Στην εικόνα 4.16.α βλέπουμε ότι το πακέτο μπορεί να αποσταλεί προς το βόρειο δρομολογητή ή προς τον πάνω δρομολογητή. Εάν ελεγχόντουσαν μόνο οι γειτονικοί σύνδεσμοι και οι δρομολογητές, το πακέτο θα μπορούσε να σταλεί μόνο προς τα επάνω. Ωστόσο, εάν η σύνδεση μεταξύ του δρομολογητή 9 και του δρομολογητή προορισμού 12 είναι ελαττωματική, πρέπει να ληφθεί μια μη ελάχιστη διαδρομή για να φτάσει το πακέτο στον προορισμό του. Προκειμένου να αποφευχθεί αυτή η κατάσταση, αρχικά, ελέγχεται η κατάσταση σφάλματος της διαδρομής NU και του δρομολογητή N (από το μηχανισμό διαχείρισης). Εάν δεν υπάρχει σφάλμα, τότε το πακέτο αποστέλλεται προς τη βόρεια κατεύθυνση. Διαφορετικά η κατεύθυνση προς τα πάνω είναι τελικά και επιλεγόμενη. Ομοίως, στην εικόνα 4.16.β, ελέγχονται οι καταστάσεις της διαδρομής NE και του δρομολογητή N και αν δεν είναι ελαττωματικές, το πακέτο αποστέλλεται προς τη βόρεια κατεύθυνση. Διαφορετικά επιλέγεται η ανατολική κατεύθυνση. Τέλος, στην εικόνα 4.16.γ, το πακέτο αποστέλλεται στην ανατολική κατεύθυνση εάν η

διαδρομή της ΕΕ και ο δρομολογητής Ε δεν είναι ελαττωματικοί. Διαφορετικά επιλέγεται η κατεύθυνση προς τα πάνω.

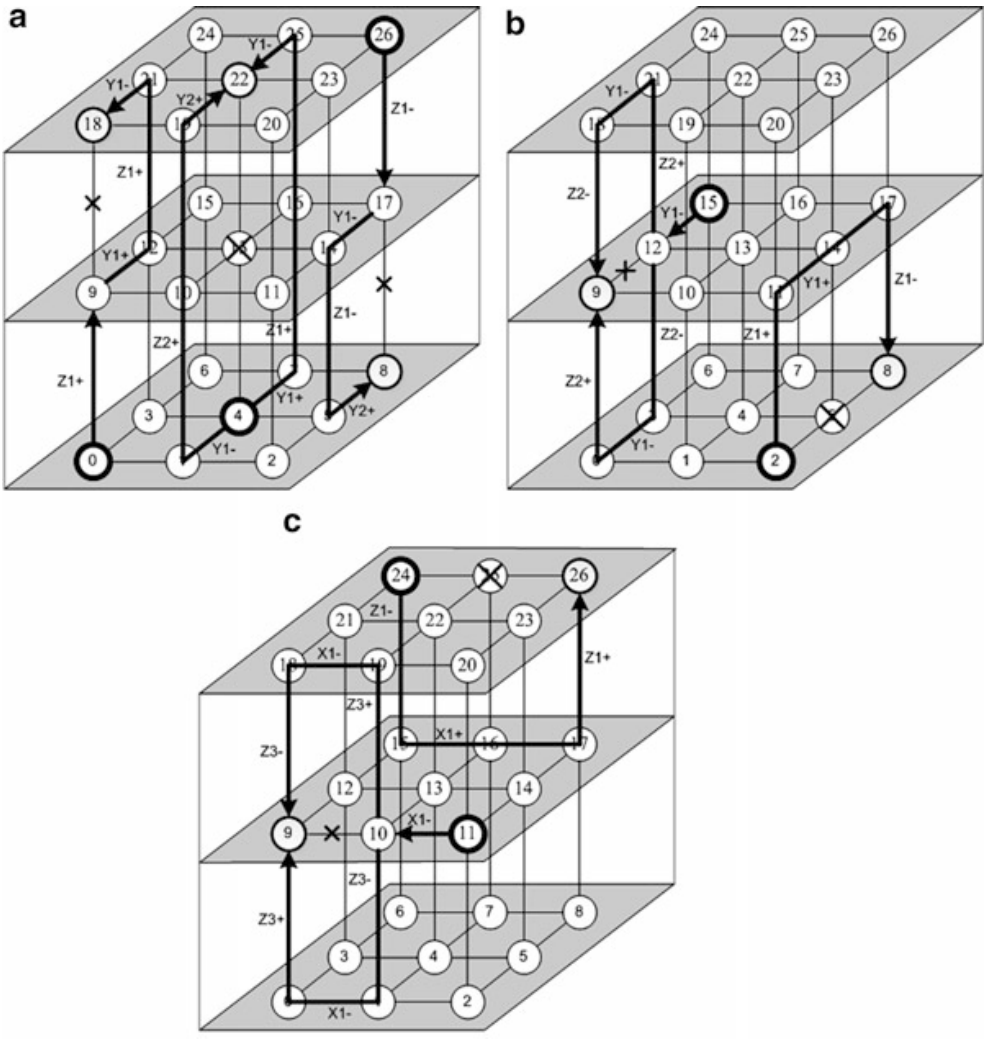
Όταν οι δρομολογητές πηγής και προορισμού βρίσκονται στην ίδια διάσταση και ένας σύνδεσμος ή δρομολογητής μεταξύ τους είναι ελαττωματικός, απαιτείται μια μη ελάχιστη διαδρομή. Για να αποφευχθεί η λήψη περιττών και μακρύτερων διαδρομών, το RR-3D προσπαθεί πάντα να μην μειώνει την απόσταση στο μηδέν κατά μήκος δύο διαστάσεων όταν η απόσταση κατά μήκος της τρίτης διάστασης είναι μεγαλύτερη από μία. Για παράδειγμα, στην εικόνα 4.16.γ, εάν η πηγή και ο προορισμός βρίσκονται στον δρομολογητή 0 και 19, αντίστοιχα, η διαθεσιμότητα της κατεύθυνσης προς τα πάνω ελέγχεται νωρίτερα από την ανατολική κατεύθυνση. Ο λόγος είναι ότι εάν το πακέτο αποστέλλεται στην ανατολική διάσταση και ο σύνδεσμος 10-19 είναι ελαττωματικός, το πακέτο πρέπει να λάβει μια ελάχιστη διαδρομή. Από την άλλη πλευρά, αποστέλλοντας το πακέτο στο δρομολογητή 9, το πακέτο έχει δύο εναλλακτικές διαδρομές για να φτάσει στον προορισμό και εάν μία από τις διαδρομές είναι ελαττωματική, το πακέτο αποστέλλεται μέσω της άλλης.

Όπως ήδη αναφέρθηκε, όταν τα πακέτα είναι east-, west-, north-, ή south- και υπάρχει ένα σφάλμα στη διαδρομή, πρέπει να ακολουθήσουν μια μη ελάχιστη διαδρομή. Το RR-3D θα πρέπει να είναι σε θέση να ανιχνεύσει και να παρακάμψει αυτούς τους ελαττωματικούς συνδέσμους. (Αξίζει να σημειωθεί ότι εάν οι δρομολογητές προέλευσης και προορισμού δεν βρίσκονται στην ίδια διάσταση, τα πακέτα δεν αντιμετωπίζουν ποτέ αυτές τις συνθήκες καθώς τα σφάλματα παρακάμπτονται πριν καν τα πακέτα μας φτάσουν στους ελαττωματικούς συνδέσμους ή δρομολογητές). Επομένως, όταν οι δρομολογητές προέλευσης και προορισμού βρίσκονται σε ίδια διάσταση, τα πακέτα ξεκινούν τη δρομολόγηση από το χαμηλότερο δυνατό υποδίκτυο και στη συνέχεια μπορούν να μεταβούν σε ένα υψηλότερο υποδίκτυο, πάντα σε

αύξουσα σειρά, αν χρειαστεί. Οι κανόνες του αλγόριθμου RR-3D έχουν ως εξής:

- Κανόνας 1: Εάν παρουσιαστεί σφάλμα στη διάσταση Z, τα πακέτα μετατοπίζονται μέσω της διάστασης Y.

- Κανόνας 2: Εάν παρουσιαστεί σφάλμα στη διάσταση X ή Y, τα πακέτα μετατοπίζονται μέσω της διάστασης Z.



Εικόνα 4.17 Ο αλγόριθμος RR-3D όταν υπάρχει μόνο μία διαδρομή μεταξύ των δρομολογητών προέλευσης και προορισμού

Για την εκτέλεση αυτών των κανόνων, τα πακέτα ενδέχεται να απαιτήσουν αλλαγή υποδικτύων στο σύστημα. Ας εξετάσουμε το παράδειγμα της Εικόνας 4.17.α όπου η πηγή και ο προορισμός βρίσκονται στον δρομολογητή 4 και 22 αντίστοιχα και ο δρομολογητής 15 είναι ελαττωματικός. Δεδομένου ότι το σφάλμα εντοπίζεται κατά μήκος της διάστασης Z, σύμφωνα με τον κανόνα 1, τα πακέτα μετατοπίζονται μέσω της διάστασης Y που μπορεί να είναι στην θετική ή αρνητική κατεύθυνση. Ας υποθέσουμε ότι η επαναδρομολόγηση πραγματοποιείται μέσω της θετικής κατεύθυνσης της διάστασης Y. Δεδομένου ότι το υποδίκτυο 1 καλύπτει το Y1+ (φαίνεται στην εικόνα 4.13), το πακέτο χρησιμοποιεί αυτό το κανάλι. Στη συνέχεια, το πακέτο πρέπει να δρομολογηθεί κατά μήκος της θετικής κατεύθυνσης της διάστασης Z. Το υποδίκτυο 1 καλύπτει επίσης το Z1+, οπότε το πακέτο εξακολουθεί να δρομολογείται χρησιμοποιώντας τα κανάλια του υποδίκτυου 1. Το πακέτο συνεχίζεται κατά μήκος αυτής της κατεύθυνσης μέχρι να φτάσει στο ίδιο στρώμα με το δρομολογητή προορισμού όπου πρέπει να σταλεί μέσω της αρνητικής κατεύθυνσης της διάστασης Y. Ωστόσο, αυτό το κανάλι δεν περιλαμβάνεται στο υποδίκτυο 1, ενώ το υποδίκτυο2 καλύπτει το Y1- έτσι ώστε το πακέτο να χρησιμοποιεί αυτό το κανάλι για να φτάσει στον δρομολογητή προορισμού. Όταν το πακέτο στέλνεται κατά μήκος της αρνητικής κατεύθυνσης της διάστασης Y στον δρομολογητή 4, το υποδίκτυο 1 δεν καλύπτει την αρνητική κατεύθυνση της διάστασης Y ενώ το υποδίκτυο 2 το καλύπτει, οπότε το πακέτο χρησιμοποιεί Y1-. Το πακέτο μπορεί να δρομολογηθεί κατά μήκος της διάστασης Z χρησιμοποιώντας το Z2+ από το ίδιο υποδίκτυο. Τέλος, η θετική κατεύθυνση της διάστασης Y δεν καλύπτεται από το υποδίκτυο 2 και το Y2+ λαμβάνεται από το υποδίκτυο 3 και το πακέτο φτάνει στον προορισμό. Καθώς κάθε δρομολογητής έχει τουλάχιστον έναν γείτονα κατά μήκος της διάστασης Y, οι ελαττωματικοί σύνδεσμοι σε κατακόρυφες συνδέσεις μπορούν να γίνουν ανεκτοί δρομολογώντας τα πακέτα μέσω της διάστασης Y.

Η Εικόνα 4.17.β δείχνει τις περιπτώσεις όπου το σφάλμα εμφανίζεται στους συνδέσμους Y και είναι ανεκτικό από την αναδρομολόγηση των πακέτων μέσω της διάστασης Z σύμφωνα με τον κανόνα 2. Εξετάζουμε το παράδειγμα όπου η πηγή και ο προορισμός βρίσκονται στους δρομολογητές 15 και 9, αντίστοιχα, και ο σύνδεσμος 12-9 είναι ελαττωματικός. Αρχικά πρέπει να ληφθεί η αρνητική κατεύθυνση της διάστασης Y. Δεδομένου ότι το υποδίκτυο 1 δεν το καλύπτει, θα ελεγχθεί το επόμενο υποδίκτυο. Το υποδίκτυο 2 καλύπτει το Y1- και έτσι το πακέτο χρησιμοποιεί αυτό το κανάλι. Για τη δρομολόγηση κατά μήκος της διάστασης Z είτε στη θετική είτε αρνητική κατεύθυνση, χρησιμοποιούνται τα κανάλια του υποδικτύου 2 (Z2*). Στη συνέχεια, το πακέτο πρέπει να δρομολογηθεί κατά μήκος της αρνητικής κατεύθυνσης της διάστασης Y που καλύπτεται από το υποδίκτυο 2 (Y1-). Τέλος, οι αρνητικές και θετικές κατευθύνσεις της διάστασης Z καλύπτονται από το ίδιο υποδίκτυο (Z2*).

Στην εικόνα 4.17.γ, σύμφωνα με τον κανόνα 2, τα ελαττώματα στη διάσταση X είναι ανεκτά από την αναδρομολόγηση των πακέτων μέσω της διάστασης Z. Ας υποθέσουμε την περίπτωση όπου η πηγή και ο προορισμός βρίσκονται στον δρομολογητή 11 και 9, αντίστοιχα, και ο σύνδεσμος 10-9 είναι ελαττωματικός. Η αρνητική κατεύθυνση της διάστασης X δεν καλύπτεται από τους υποδίκτυα 1 και 2 και επομένως χρησιμοποιείται το X1- από το υποδίκτυο 3. Το πακέτο μετατοπίζεται κατά μήκος της διάστασης Z χρησιμοποιώντας Z3+ ή Z3- από το ίδιο υποδίκτυο. Το πακέτο πρέπει να περάσει μέσω του X1- και στη συνέχεια μέσω του Z3 + ή Z3- για να φτάσει στο δρομολογητή προορισμού. Όλα αυτά τα κανάλια καλύπτονται από το υποδίκτυο 3. Επειδή κάθε δρομολογητής έχει τουλάχιστον ένα γείτονα στη διάσταση Z, οι ελαττωματικοί σύνδεσμοι στη διάσταση X ή Y μπορούν να γίνουν ανεκτοί μεταβάλλοντας τα πακέτα κατά μήκος της διάστασης Z.

4.4 Αποτελέσματα προσομοίωσης

Για να αξιολογηθεί η αποτελεσματικότητα των προτεινόμενων αλγορίθμων δρομολόγησης, έχει αναπτυχθεί ένας προσομοιωτής 2D και 3D NoC με VHDL για να μοντελοποιηθούν όλα τα κύρια συστατικά του δικτύου on-chip. Για όλους τους δρομολογητές, το πλάτος δεδομένων είναι ρυθμισμένο στα 32 bit. Κάθε buffer εισόδου μπορεί να φιλοξενήσει έξι flits σε κάθε κανάλι. Επιπλέον, το μήκος του πακέτου κατανέμεται ομοιόμορφα μεταξύ 1 και 8 flits. Το ποσοστό αιτήσεων ορίζεται ως ο λόγος της επιτυχούς injection πακέτου στο δίκτυο σε σχέση με τον συνολικό αριθμό των προσπαθειών injection. Ως μέτρηση επιδόσεων, χρησιμοποιούμε τον latency που ορίζεται ως ο αριθμός κύκλων μεταξύ της εισαγωγής του πακέτου στο σύστημα μέχρι την εξαγωγή του από αυτό.

Για την αξιολόγηση της απόδοσης σε ένα δίκτυο 2D mesh, ο αλγόριθμος RR-2D συγκρίνεται με μια επαναρυθμίσιμη δρομολόγηση [15] (μέθοδο ReRS). Το ReRS δεν απαιτεί κανένα εικονικό κανάλι και είναι σε θέση να ανεχτεί όλες τις θέσεις ενός και μόνο ελαττωματικού δρομολογητή. Ωστόσο, χρησιμοποιώντας το ReRS, χρειάζεται να δεχτούμε και περιττές διαδρομές (μεγαλύτερες της ελάχιστης) για να ανεχθούν σφάλματα που έχουν ως αποτέλεσμα τη δημιουργία συμφόρησης γύρω από τις ελαττωματικές περιοχές. Επιπλέον, το ReRS βασίζεται στην ντετερμινιστική δρομολόγηση και έτσι τα πακέτα δεν μπορούν να διανεμηθούν ομοιόμορφα στο δίκτυο. RR-2D βασίζεται σε μια πλήρως προσαρμοστική δρομολόγηση και χρησιμοποιεί ένα και δύο εικονικά κανάλια κατά μήκος των διαστάσεων X και Y, αντιστοίχως. Το RR-2D είναι σε θέση να ανεχτεί όλες τις θέσεις ενός μόνο ελαττωματικού συνδέσμου ή δρομολογητή. Για να έχουμε καλύτερη εικόνα στη σύγκριση χρησιμοποιούμε τον ίδιο αριθμό εικονικών καναλιών και στις δύο μεθόδους και για το σκοπό αυτό προστίθεται ένα επιπλέον εικονικό

κανάλι στην προσέγγιση ReRS. Αυτό το εικονικό κανάλι χρησιμοποιείται για σκοπούς απόδοσης.

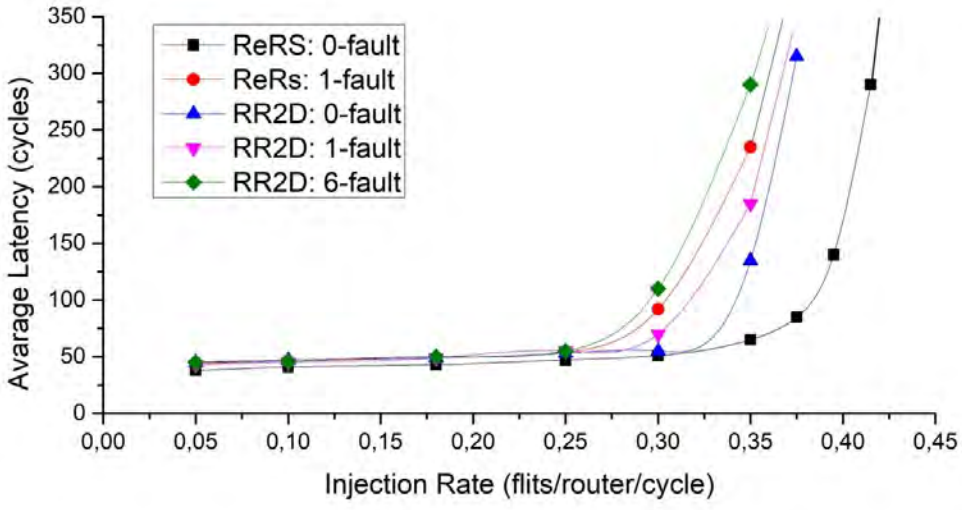
Για τη μέτρηση της απόδοσης σε δίκτυο 3D mesh, η μέθοδος (RR-3D) συγκρίνεται με το HamFA [16]. Το HamFA είναι μια μέθοδος fault-tolerant. Είναι σε θέση να ανεχτεί σφάλματα είτε σε κάθετη είτε σε οριζόντια σύνδεση χωρίς χρήση εικονικών καναλιών. Το HamFA είναι ένας μερικώς προσαρμοστικός αλγόριθμος δρομολόγησης. Από την άλλη πλευρά, το RR-3D μπορεί να ανεχτεί και ελαττωματικούς συνδέσμους και ελαττωματικούς δρομολογητές ενώ ταυτόχρονα εγγυάται την ανεκτικότητα όλων των σφάλματος οπουδήποτε στο δίκτυο. Το RR-3D βασίζεται σε έναν πλήρως προσαρμοστικό αλγόριθμο δρομολόγησης που απαιτεί δύο, δύο και τέσσερα εικονικά κανάλια. Για να υπάρχει ισοκατανομή στη σύγκριση, ο ίδιος αριθμός εικονικών καναλιών χρησιμοποιείται και στις δύο μεθόδους.

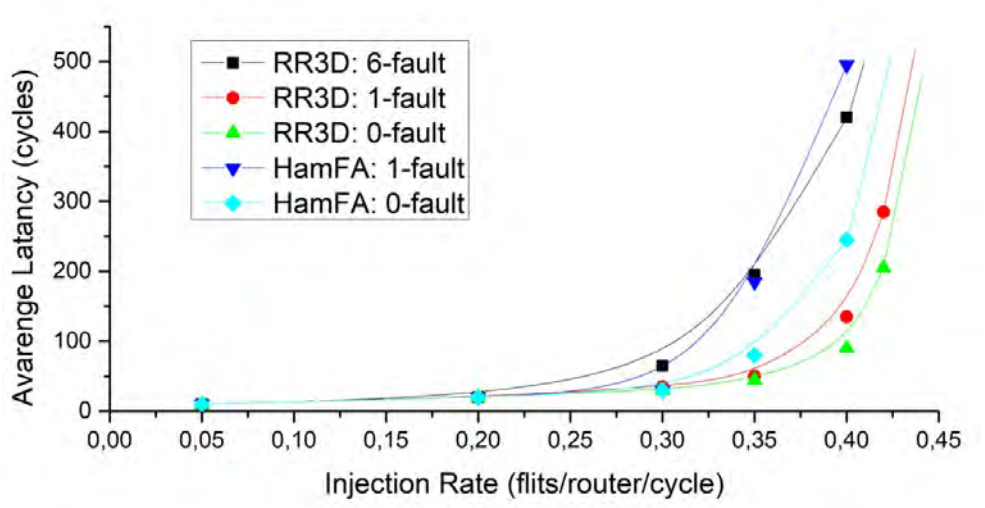
Εκτελούμε δύο σειρές προσομοιώσεων: 1- για να μετρήσουμε την απόδοση των προτεινόμενων μεθόδων έναντι των βασικών μεθόδων σε δίκτυα 2D και 3D mesh. 2- να μετρηθεί η αξιοπιστία της προτεινόμενης μεθόδου σε σύγκριση με τις βασικές μεθόδους. Και στις δύο σειρές προσομοιώσεων εκτελούμε τα πειράματα σε ένα δίκτυο 8×8 (για 2D) και 4×4×4 (για 3D) mesh. Για ανάλυση απόδοσης, ο προσομοιωτής warmed up για 20.000 κύκλους και στη συνέχεια μετράται η μέση απόδοση σε άλλους 200.000 κύκλους.

4.4.1 Ανάλυση απόδοσης κάτω από Uniform Traffic Profile

Στο Uniform Traffic Profile, κάθε στοιχείο επεξεργασίας δημιουργεί πακέτα δεδομένων και τα στέλνει σε άλλο στοιχείο επεξεργασίας χρησιμοποιώντας μια ομοιόμορφη κατανομή [17]. Στην εικόνα 4.18.α, η μέση λανθάνουσα επικοινωνίες των RR-2D και ReRS μετρούνται με και χωρίς σφάλματα και για μεμονωμένες περιπτώσεις σφαλμάτων δρομολογητή. Επιπλέον, η

απόδοση του RR-2D μετράται κάτω από έξι σφάλματα στο δίκτυο, με τυχαία επιλεγμένους ελαττωματικούς συνδέσμους ή δρομολογητές. Όπως παρατηρήθηκε από τα αποτελέσματα, σε περιπτώσεις σφαλμάτων, η μέθοδος ReRS εκτελείται το καλύτερα, καθώς βασίζεται σε ντετερμινιστική δρομολόγηση (δηλαδή παρόμοια με τη δρομολόγηση XY) η οποία είναι κατάλληλη για ομοιόμορφη κυκλοφορία. Όταν εμφανίζεται ένα μόνο σφάλμα στο δίκτυο, η απόδοση της μεθόδου ReRS μειώνεται σημαντικά ενώ η μέθοδος RR-2D διατηρεί την απόδοση παρουσία σφαλμάτων, ακόμη και αν υπάρχουν έξι σφάλματα στο δίκτυο. Ο λόγος είναι ότι όχι μόνο η RR-2D αποφεύγει τη λήψη περιττών μη ελάχιστων διαδρομών, αλλά επίσης βασίζεται σε μια πλήρως προσαρμοστική δρομολόγηση και επομένως αποδίδει καλά στην διανομή πακέτων σε διαφορετικές διαδρομές.



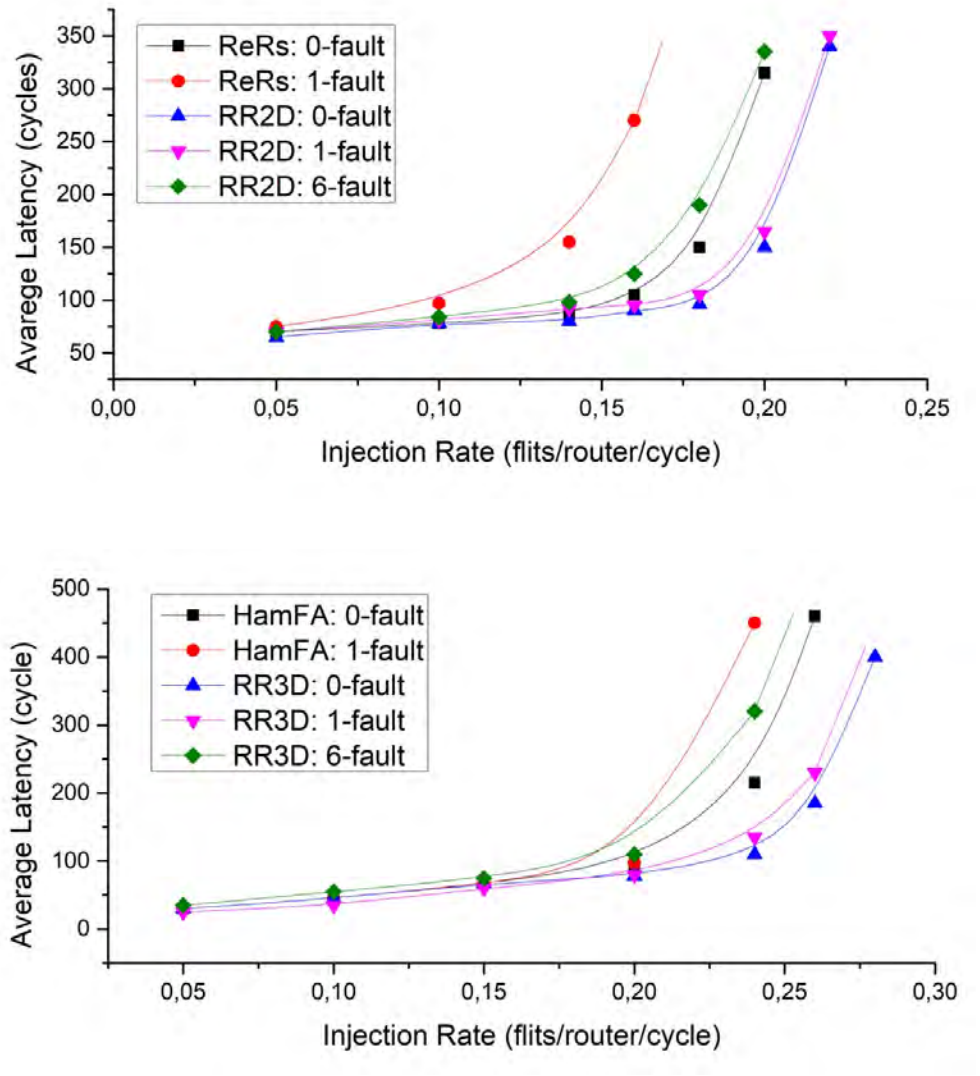


Εικόνα 4.18 Απόδοση με uniform traffic profile σε (α) 2D δίκτυο, (β) 3D δίκτυο

4.4.2 Ανάλυση απόδοσης σε Hotspot Traffic Profile

Κάτω από το πρότυπο κυκλοφορίας hotspot, ένας ή περισσότεροι δρομολογητές επιλέγονται ως hotspots, λαμβάνοντας επιπλέον μέρος της κυκλοφορίας εκτός από την κανονική ομοιόμορφη κίνηση. Δεδομένου ενός ποσοστού hotspot του H, ένα πακέτο που δημιουργήθηκε πρόσφατα κατευθύνεται σε κάθε δρομολογητή hotspot με επιπλέον πιθανότητα H%. Υπολογίζουμε την κυκλοφορία hotspot με H = 10% σε έναν δρομολογητή hotspot στο (4,4) σε ένα δίκτυο NoC 2D 8×8 και δύο δρομολογητές hotspot στις θέσεις (2,1,2) και (3,2,2) σε ένα 4×4×4 NoC 3D, αντίστοιχα. Στην εικόνα 19 α), οι επιδόσεις των RR-2D και ReRS μετριούνται για χωρίς σφάλματα και με σφάλμα ενός δρομολογητή, ενώ η απόδοση του RR-2D και κάτω από έξι σφάλματα (με μίξη ελαττωματικών δρομολογητών και συνδέσμων). Το RR-2D έχει πολύ καλύτερη απόδοση από το ReRS, ακόμη και υπό την παρουσία έξι σφαλμάτων. Αυτό οφείλεται στο γεγονός ότι το RR-2D είναι μια προσαρμοστική μέθοδος και μπορεί να εξισορροπήσει την κυκλοφορία μέσω του δικτύου. Από την άλλη πλευρά, το RR-2D αποφεύγει τη χρήση μη-

ελάχιστων διαδρομών όσο δυνατόν περισσότερο. Η απόδοση των μεθόδων RR-3D και HamFA απεικονίζεται στην εικόνα 19 β). Όπως παρατηρείται από αυτό το σχήμα, το RR-3D οδηγεί στην καλύτερη απόδοση σε περιπτώσεις χωρίς σφάλματα. Όταν υπάρχουν έξι σφάλματα στο δίκτυο, η απόδοση εξακολουθεί να είναι υψηλή και σημαίνει ότι το RR-3D έχει καλές επιδόσεις στις ανοχές κατά τη διατήρηση της απόδοσης.

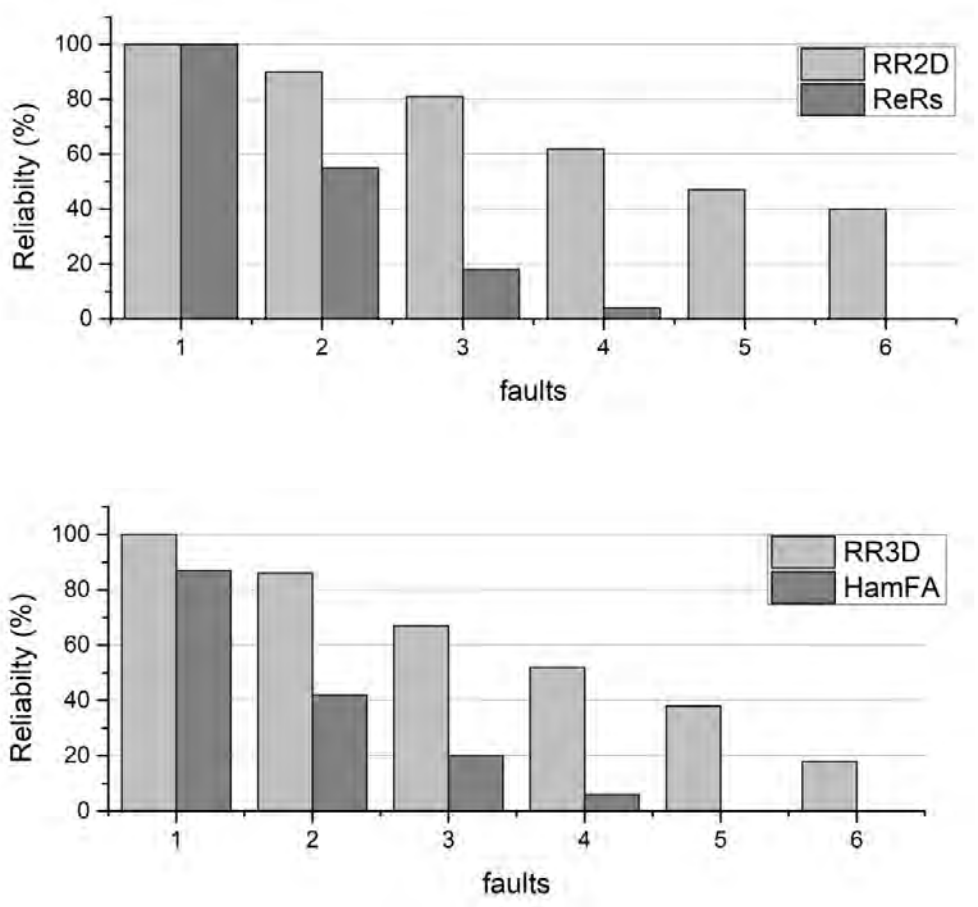


Εικόνα 4.19 Απόδοση σε hotspot traffic profile σε α) 2D δίκτυο, β) 3D δίκτυο

4.4.3 Αξιολόγηση αξιοπιστίας σε Uniform Traffic Profile

Καθώς το ReRS είναι σε θέση να ανεχτεί ελαττωματικούς δρομολογητές και συνδέσμους στη βασική του μορφή, μετράμε την αξιοπιστία του απενεργοποιώντας μόνο τους δρομολογητές. Το RR-2D μπορεί να ανεχτεί και τους δύο σφαλμένους δρομολογητές και τους συνδέσμους και η αξιοπιστία μετράται με το μείγμα αυτών. Αυξάνουμε τον αριθμό των σφαλμάτων από 1 έως 6. Όλα τα σφάλματα επιλέγονται με τυχαία λειτουργία. Ένα δίκτυο είναι αξιόπιστο αν όλα τα πακέτα που εισέρχονται σε αυτό φτάνουν στους προορισμούς τους. Με άλλα λόγια, το δίκτυο θεωρείται αναξιόπιστο ακόμα και αν όλα τα πακέτα φτάσουν στους προορισμούς εκτός από ένα πακέτο. Όπως φαίνεται στην εικόνα 4.20.α, ο RR-2D μπορεί να αντέξει έως και έξι σφάλματα με αξιοπιστία μεγαλύτερη από 40%.

Σε ένα δίκτυο 3D, συγκρίνουμε την αξιοπιστία του RR-3D με το HamFA. Καθώς το HamFA έχει σχεδιαστεί για να αντέχει ελαττωματικούς συνδέσμους μονής κατεύθυνσης, η αξιοπιστία του μετράται επίσης με βάση τη συγκριμένη επιλογή. Από την άλλη πλευρά, το RR-3D αντέχει τόσο τους ελαττωματικούς συνδέσμους όσο και τους δρομολογητές και η τιμή αξιοπιστίας επιτυγχάνεται υπό την παρουσία και των δύο ειδών βλαβών. Εισάγουμε 1 έως 6 σφάλματα στο δίκτυο (μείγμα ελαττωματικών δρομολογητών και συνδέσμων) για να μετρήσουμε την αξιοπιστία του RR-3D, ενώ εισάγονται 1 έως 6 μονολιθικοί ελαττωματικοί σύνδεσμοι για να μετρηθεί η αξιοπιστία του HamFA. Όπως φαίνεται στην εικόνα 4.20.β, η αξιοπιστία του HamFA μειώνεται σημαντικά σε σύγκριση με την RR-3D. Χρησιμοποιώντας τον RR-3D με πιθανότητα 18%, το δίκτυο λειτουργεί κανονικά χωρίς απώλεια πακέτων όταν υπάρχουν έξι σφάλματα στο δίκτυο.



Εικόνα 20 Αξιολόγηση αξιοπιστίας σε α) Δίκτυο 2D, β) Δίκτυο 3D

4.5 Συμπεράσματα

Σε αυτό το κεφάλαιο, είδαμε μεθόδους ανθεκτικές σε σφάλματα υψηλής απόδοσης για 2D και 3D NoCs. Αυτές οι μέθοδοι βασίζονται σε πλήρως προσαρμοστικούς αλγόριθμους, αποφεύγοντας έτσι τη συμφόρηση στο δίκτυο. Οι προτεινόμενοι αλγόριθμοι ανθεκτικότητας σε σφάλματα προτιμούνται για την ανοχή τους σε ελαττωματικούς δρομολογητές και συνδέσμους. Χρησιμοποιούν μόνο τα συντομότερα μονοπάτια για να ανεχθούν σφάλματα, εφόσον υπάρχει τέτοια διαδρομή. Οι μη ελάχιστες διαδρομές είναι απαραίτητες όταν η πηγή και ο προορισμός βρίσκονται στην ίδια διάσταση και υπάρχει σφάλμα μεταξύ τους. Αυτά τα είδη βλαβών είναι

ανεκτά χωρίς τη δημιουργία κύκλων στο δίκτυο. Σε ένα δίκτυο 3D, παρουσιάσαμε έναν πλήρως προσαρμοστικό αλγόριθμο δρομολόγησης με τη διαίρεση του δικτύου σε τέσσερα διαφορετικά υποδίκτυα. Η δυνατότητα σφάλματος βελτιώνεται επιτρέποντας στα πακέτα να μετακινούνται μεταξύ υποδικτύων με την αυστηρά αύξουσα σειρά.

Βιβλιογραφία

1. M. Moadeli, A. Shahrabi, W. Vanderbauwhede, M. Ould-Khaoua, An analytical performance model for the Spidergon NoC, in *Proceedings of 21st Annual Conference on Advanced Networking and Applications* (2007), pp. 1014–1021

2. Y. Xie, G.H. Loh, B. Black, K. Bernstein, Design space exploration for 3D architectures. ACM J. Emerg. Technol. Comput. Syst. (JETC) **2**(2), 65–103 (2006)

3. M. Daneshtalab, M. Ebrahimi, P. Liljeberg, J. Plosila, H. Tenhunen, High-performance TSV architecture for 3-D ICs, in *Proceedings of IEEE Computer Society Annual Symposium on VLSI (ISVLSI)* 2010, pp. 467–468

4. A.Y. Weldezion, M. Grange, D. Pamunuwa, Z. Lu, A. Jantsch, R. Weerasekera, H. Tenhunen, Scalability of network-on-chip communication architecture for 3-D meshes, in *Proceedings of 3rd ACM/IEEE International Symposium on Networks-on-Chip (NOCS)* (2009), pp. 114–123

5. J. Hu, L. Wang, L. Jin, H.Z. JiangNan, Electrical modeling and characterization of through silicon vias (TSV), in *Proceedings of International Conference on Microwave and Millimeter Wave Technology (ICMMT)*, vol. 2 (2012), pp. 1–4

6. B.S. Feero, P.P. Pande, Networks-on-chip in a three-dimensional environment: A performance evaluation. IEEE Trans. Comput. **58**(1), 32–45 (2009)

7. C. Seiculescu, S. Murali, L. Benini, G. De Micheli, SunFloor 3D: A tool for networks on chip topology synthesis for 3D systems on chips, in *Proceedings of Design, Automation Test in Europe Conference Exhibition (DATE)* (2009), pp. 9–14

8. H. Matsutani, M. Koibuchi, H. Amano, Tightly-coupled multi-layer topologies for 3-D NoCs, in *Proceedings of 41st International Conference on Parallel Processing* (2007), p. 75

9. V.F. Pavlidis, E.G. Friedman, 3-D topologies for networks-on-chip. IEEE Trans. VLSI Syst. **15**(10), 1081–1090 (2007)

10. F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, M.K, Design and management of 3D chip multiprocessors using network-in-memory, in *Proceedings of ISCA-33* (2006), pp. 130–141

11. W. Dally, B. Towles, *Principles and Practices of Interconnection Networks* (Morgan Kaufmann, San Francisco, 2003)

12. L.M. Ni, P.K. McKinley, A survey of wormhole routing techniques in direct networks. Computer **26**(2), 62–76 (1993)

13. C.J. Glass, C.J. Glass, L.M. Ni, L.M. Ni, Maximally fully adaptive routing in 2D meshes, in *Proceedings of International Conference on Parallel Processing* (1992), pp. 101–104

14. M. Ebrahimi, X. Chang, M. Daneshtalab, J. Plosila, P. Liljeberg, H. Tenhunen, DyXYZ: Fully adaptive routing algorithm for 3D NoCs, in

Proceedings of 21th IEEE Euromicro Conference on Parallel, Distributed and Network-Based Computing (PDP) (2013), pp. 499–503

15. Z. Zhang, A. Greiner, S. Taktak, A reconfigurable routing algorithm for a fault-tolerant 2D-mesh network-on-chip, in *Proceedings of 45th ACM/IEEE Design Automation Conference (DAC)* (2008), pp. 441–446

16. M. Ebrahimi, M. Daneshtalab, J. Plosila, Fault-tolerant routing algorithm for 3D NoC using Hamiltonian path strategy, in *Proceedings of 16th ACM/IEEE Design, Automation, and Test in Europe (DATE)* (2013), pp. 1601–1605

17. C.J. Glass, L.M. Ni, The turn model for adaptive routing, in Proceedings of the 19th Annual International Symposium on Computer Architecture (1992), pp. 278–287

$5^{ο}$ ΚΕΦΑΛΑΙΟ

ΓΕΝΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ ΣΕ ΣΥΣΤΗΜΑΤΑ NOC

5.1 Εισαγωγή

Τα μελλοντικά συστήματα on-chip (SoC) θα αποτελούνται από εκατοντάδες επεξεργαστές και στοιχεία μνήμης ικανά να μεταφέρουν Gigabytes δεδομένων ανά δευτερόλεπτο. Η αρχιτεκτονική τους θα παίξει καθοριστικό παράγοντα στην απόδοση και στην συνολική κατανάλωση ισχύος των συγκεκριμένων συστημάτων. Τα NoC έχουν αναδειχθεί ως η κυρίαρχη λύση στην αρχιτεκτονική διασύνδεσης των SoC (σε νανοκλίμακα) τόσο από ακαδημαϊκής πλευράς [1] [2], όσο και από βιομηχανικής [3].

5.2 Ταξινόμηση Γενετικών Αλγορίθμων

Μεταχειρητική (Metaheuristics) είναι μια διαδικασία που στοχεύει στην εξεύρεση αποδεκτής λύσης σε πολύπλοκα προβλήματα βελτιστοποίησης και αναζήτησης. [4] Χρησιμοποιεί συγκεκριμένες μεθόδους εύρεσης οι οποίες χρησιμοποιούνται σε αλγορίθμους που είναι πιο αφηρημένοι. Σε σύγκριση με του αλγορίθμους βελτιστοποίησης και τις επαναληπτικές μεθόδους, η ανακτημένη λύση εξαρτάται από το σύνολο των τυχαίων μεταβλητών που δημιουργούνται.

Οι γενετικοί αλγόριθμοι έχουν τα ακόλουθα χαρακτηριστικά:

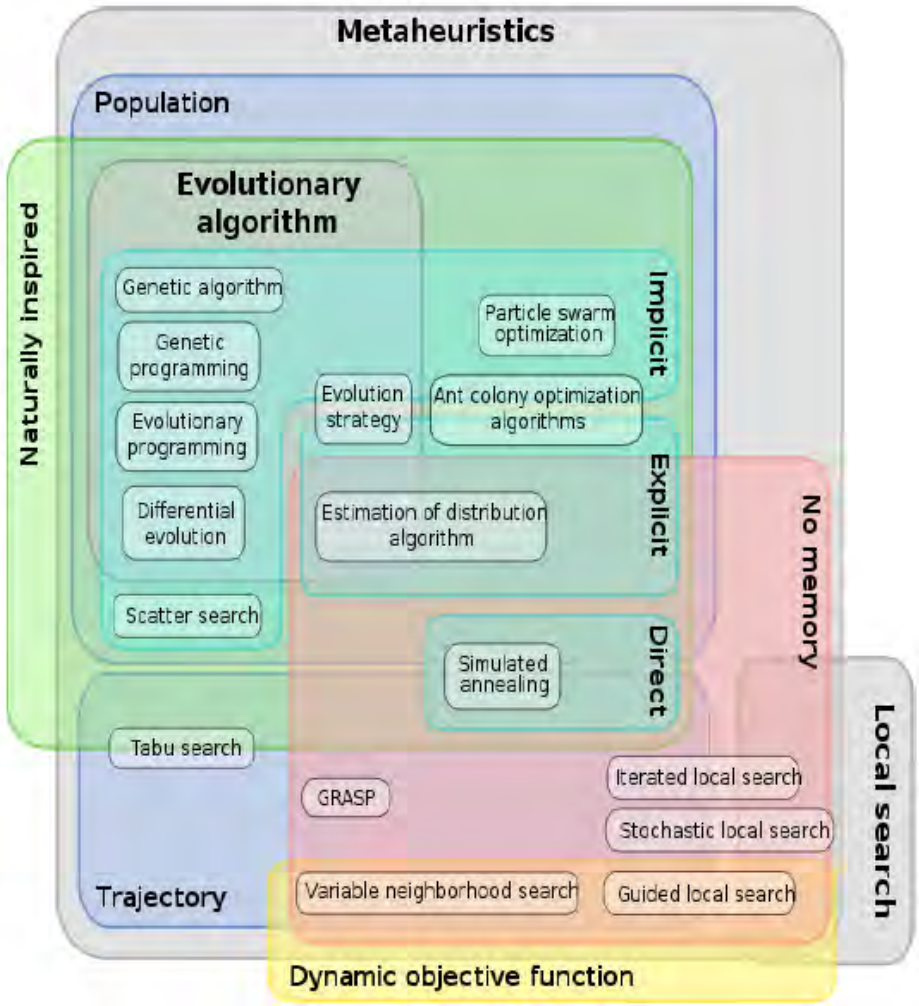
- Το metaheuristicks καθοδηγεί τη διαδικασία αναζήτησης με συγκεκριμένες στρατηγικές. Αν και τυχαίοι, μεταχειρητικοί αλγόριθμοι όπως οι GA δεν κάνουν τυχαίες αναζητήσεις. Χρησιμοποιούν τις προηγούμενες πληροφορίες για να κατευθύνουν την αναζήτηση σε μια περιοχή με καλύτερες επιδόσεις στο χώρο της αναζήτησης.

- Στόχος τους είναι να κάνουν αποτελεσματική αναζήτηση στο χώρο για να βρουν βέλτιστες λύσεις. Έτσι οι μεταχειρητικές αναζητήσεις περιλαμβάνουν απλούς αλγορίθμους με σύνθετες διαδικασίες εκμάθησης.

- Οι μεταχειρητικοί αλγόριθμοι είναι στην πλειονότητά τους μη ντετερμινιστικοί.

- Οι μεταχειρητικοί αλγόριθμοι χαρακτηρίζονται ως non-problem-spesific. Κάνουν λίγες υποθέσεις σχετικά με το πρόβλημα βελτιστοποίησης ή αναζήτησης που πρέπει να επιλυθεί και επομένως μπορούν να χρησιμοποιηθούν σε μια μεγάλη γκάμα προβλημάτων.

Οι GA κάνουν πληθυσμιακές αναζητήσεις με μαθησιακές συνιστώσες.

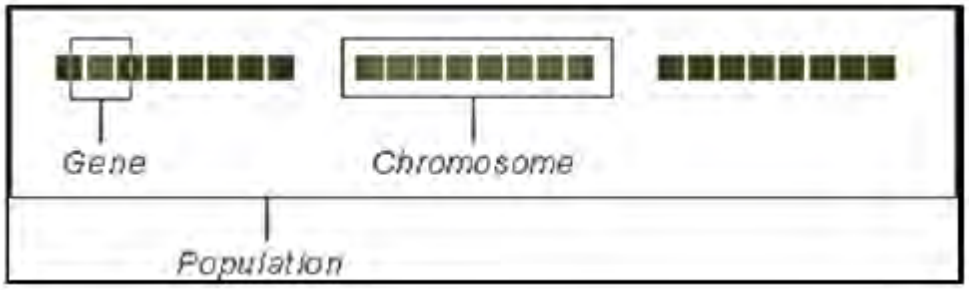


Εικόνα 5.1 Ταξινόμηση χαρακτηριστικών μεταχειρητικών αλγορίθμων [5]

5.2.1 Αντιπροσώπευση Γενετικών αλγορίθμων με τη χρήση bit

Για να αντιπροσωπευθούν τα χρωμοσώματα σε έναν πληθυσμό γενετικών αλγορίθμων χρησιμοποιούνται ακολουθίες δυαδικών ψηφίων παρόλο που υπάρχουν πολλές άλλες μέθοδοι οι οποίες ακόμα βρίσκονται σε πρώιμο στάδιο και είναι υπό εξέλιξη όπως η fix-point binary representation, η order-based representation, η embedded list, η variable element list και η LISP S-expression.

Κάθε θέση στη συμβολοσειρά μπορεί να πάρει μόνο δύο τιμές την ‘0’ και την ‘1’. Μια λύση (χρωμόσωμα) μπορεί να αποτελείται από διάφορες μεταβλητές (γονίδια). Ένα χρωμόσωμα αντιπροσωπεύει μια λύση από το σύνολο των πιθανών λύσεων στο χώρο αναζητήσεως. Ένας πληθυσμός περιέχει ένα σύνολο χρωμοσωμάτων που αποτελούνται από γονίδια. Με άλλα λόγια, ο χώρος λύση περιέχει ένα σύνολο συμβολοσειρών αποτελούμενες από bits.



Εικόνα 5.2 Πληθυσμός, χρωμοσώματα και γονίδια

Ο αρχικός πληθυσμός παράγεται τυχαία. Στη συνέχεια ο GA περνά στους ακόλουθους διαχειριστές:

α) Διαχειριστής επιλογής

Η αρχή που ισχύει εδώ είναι “επιβιώνει ο πιο ικανός”. Επιλέγονται χρωμοσώματα από τον πληθυσμό για αναπαραγωγή. Με βάση μια λειτουργία αξιολόγησης, επιλέγονται κάποια χρωμοσώματα και αυξάνετε η πιθανότητά τους να επιλεγούν για αναπαραγωγή. Η υλοποίησή του εξαρτάτε από το πρόβλημα που έχουμε για επίλυση. Τα καλύτερα “χρωμοσώματα” μπορούν να προσδιοριστούν από μια αντικειμενική λειτουργία που εφαρμόζεται ομοιόμορφα σε όλα τα χρωμοσώματα ή από μια υποκειμενική όπου μερικοί κανόνες μπορεί να μην εφαρμόζονται ομοιόμορφα. Είναι πιθανόν το

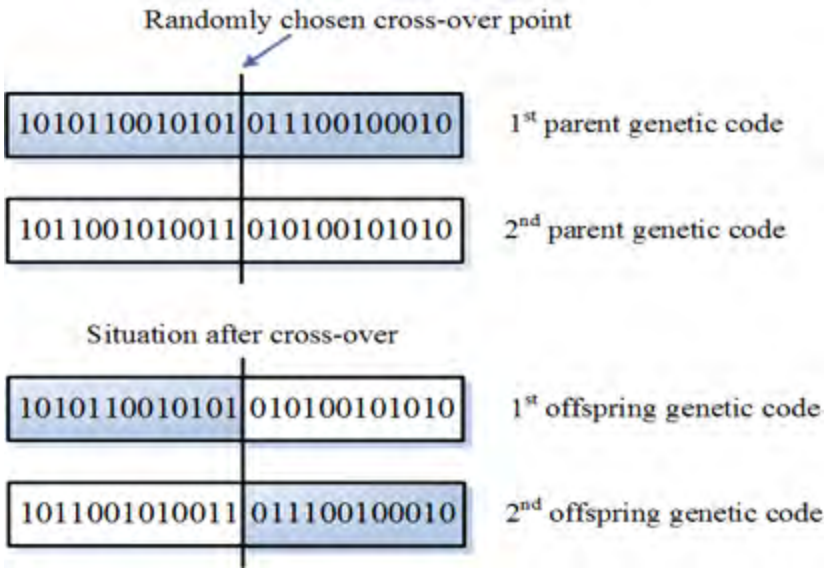
καλύτερο χρωμόσωμα να μην επιλεγεί αν εκτελεστεί ο GA, αφού όταν ο αλγόριθμος εκτελεστεί αρκετές φορές η πιθανότητα επιλογής συγκλίνει προς τη μαθηματική του αναμενόμενη τιμή.

β) Διαχειριστής διασταύρωσης

Αυτός ο χειριστής επιλέγει τυχαία μια θέση στη συμβολοσειρά. Στη συνέχεια, οι ακολουθίες πριν και μετά από αυτή τη θέση ανταλλάσσονται μεταξύ των συμβολοσειρών. Το αποτέλεσμα είναι η δημιουργία δύο νέων χρωμοσωμάτων. Παίρνουμε για παράδειγμα δύο συμβολοσειρές που αντιπροσωπεύουν χρωμοσώματα π.χ. 11000 και 00111. Ένα διαχειριστής διασταύρωση επιλέγει τυχαία μια θέση π.χ. τη δεύτερη. Τα προκύπτοντα απιδιά του διασταυρωτή θα είναι τα 11111 και 00000.

Parents: 11|000 και **00|111**

Children: 11111 και **00000**



Εικόνα 5.3 Γενετικός κώδικας γονέων και παιδιών μετά τη διασταύρωση [6]

Οι διασταυρώσει πολλαπλών σημείων είναι απλά διαγραμμίσεις με περισσότερες από μια θέσεις όπου θα υπάρξει διασταύρωση.

γ) Χειριστής μεταλλάξεων

Αυτή η διαδικασία μιμείται την ποικιλομορφία σε έναν πληθυσμό και βοηθά στην πρόληψη της πρόωρης σύγκλισης. Αυτό συμβαίνει αφού μόνο η μετάλλαξη και η επιλογή δημιουργούν αλγορίθμους hill-climbing. Ο χειριστής αλλάζει τυχαία ένα ή μερικά bit στην συμβολοσειρά ενός χρωμοσώματος. Για παράδειγμα μια σειρά 10101010 θα μπορούσε να μεταλλαχθεί στην πρώτη θέση για να δώσει 00101010. Η πιθανότητα και η θέση μετάλλαξης ελέγχονται από τον χειριστή.

Before mutation: 10101010

After mutation: 00101010

δ) Χειριστής αναστροφής

Αυτός ο χειριστής αλλάζει τη σειρά γονιδίων μεταξύ δύο τυχαία επιλεγμένων σημείων σε ένα χρωμόσωμα. Αυτός ο χειριστής χρησιμοποιείται σπάνια καθώς τα πλεονεκτήματά του είναι ελάχιστα.

5.3 Αρχές Bio-Inspired αλγορίθμων στα NoC

Τα noc μπορούν να θεωρηθούν και ως ένα μικροδίκτυο του οποίου η σύνδεση OCI (On Chip – Interconnect) των στοιχείων του είναι ο θεμέλιος λίθος μεταξύ των PEs και των πυρήνων. Σε κάθε επίπεδο, απαιτούνται μηχανισμοί παρακολούθησης και εποπτείας για την αναστολή των self-* χαρακτηριστικών του. Χρησιμοποιώντας το μοντέλο stack micronetwork οι δυνατότητες των self-* μπορούν να οριστούν ως εξής (βλέπε πίνακα 5.1). Αυτό επιτρέπει στα συστατικά των ανώτερων επιπέδων να εποπτεύουν αυτά

του χαμηλότερου. Η εποπτεία αυτή επικεντρώνεται στα PEs και προσαρμόζονται ανάλογα με τις διεργασίες του συστήματος. [7]

BNOC	Self-* Capabilities	Immune System
Application Level - Mapping - Scheduling	Self – Configuration - Adapt to changes	Innate level - Engulf pathogens - Activate T-Cells
Communication Level - Routing - Switching - Flow Control	Self – healing - Act to the disruption	Adaptive level (T-Cells) - Learn Ag structure - Activate B-Cells
Architecture level - Buffer - Bandwidth - Link	Self-optimizing - Tune resources	Adaptive level (B-Cells) - Learn Ag behavior - Produce antibodies

Πίνακας 5.1 Δυνατότητες self-*

Όπως και στο ανοσοποιητικό σύστημα, θεωρούμε τρείς τύπους παραγόντων, τον P-παράγοντα, τον T-παράγοντα και τον B-παράγοντα. Σε επίπεδο εφαρμογής τα έμφυτα ανοσολογικά συστατικά πρέπει να είναι αναδιαμορφωτικά. Παρόμοια με τα φαγοκύτταρα όπου μετατρέπουν τα παθογόνα σε αντιγόνα και τα "προωθούν" στο ανοσοποιητικό σύστημα (T-κύτταρα), έτσι και οι P-παράγοντες αναλύουν τις απαιτήσεις της εφαρμογής, τις αποσυνθέτουν και τις χαρτογραφούν προωθώντας τες στους T-παράγοντες. Επιπλέον οι P-παράγοντες θα πρέπει να προσαρμόζονται δυναμικά στις αλλαγές του περιβάλλοντος, όπως αλλαγές στα χαρακτηριστικά του συστήματος ή στις αλλαγές συμπεριφοράς του συστήματος με ανάπτυξη νέων διεργασιών. Αυτή η δυναμική προσαρμογή συμβάλλει στη συνεχή λειτουργία του συστήματος υπό άγνωστες συνθήκες.

Σε επίπεδο επικοινωνίας, με μηχανισμό παρόμοιο αυτού των T-κυττάρων, όπου συντονίζουν και ρυθμίζουν τα B-κύτταρα, έτσι και οι T-παράγοντες είναι υπεύθυνα για την περιφερειακή παρακολούθηση, την κατανομή και τον προγραμματισμό των εργασιών των B-παραγόντων επιλέγοντας την απαιτούμενη τεχνική δρομολόγησης/ελέγχου ροής για την διαμόρφωση του

δικτύου. Οι Τ-παράγοντες έχουν την ικανότητα να αυτό-θεραπεύονται κάτι που τους επιτρέπει να ανακαλύψουν, να διαγνώσουν και να αντιδράσουν σε καταστάσεις όπως αυτή ενός ενδεχόμενου deadlock. Κάτι τέτοιο θα μπορούσε να συμβεί, αν για παράδειγμα, μια ομάδα δρομολογητών είναι αποκλεισμένη και δεν μπορεί να προχωρήσει η ροή αν πρώτα δεν ελευθερωθεί το buffer η κάποια κανάλια. Σε αυτές τις περιπτώσεις οι Τ-παράγοντες παρακολουθούν και μεταδίδουν τις πληροφορίες αυτές σε όλους τους άλλους Τ-παράγοντες για να γνωρίζουν όλοι τους κρίσιμους κόμβους του δικτύου και να κατευθύνουν ανάλογα τη ροή των δεδομένων. Για παράδειγμα με βάση τις μεταβαλλόμενες απαιτήσεις επικοινωνίας του συστήματος οι Τ-παράγοντες μπορούν να αλλάξουν δυναμικά (μεταξύ πακέτου και αλλαγής κυκλώματος).

Στο κομμάτι της αρχιτεκτονικής, όμοια με τα Β-κύτταρα που είναι ικανά να αλλάζουν την συμπεριφορά τους ώστε να προσαρμόζονται στη συμπεριφορά των αντιγόνων, έτσι και οι Β-παράγοντες εφαρμόζουν διάφορες τεχνικές για τον συντονισμό των πόρων, την κατανομή του buffer και τον καταμερισμό του εύρους ζώνης. Με άλλα λόγια ο Β-παράγοντες έχουν την δυνατότητα αυτό-βελτίωσης που τους επιτρέπει να παρακολουθούν και να συντονίζουν αυτόματα τους πόρους του συστήματος με στόχο την αύξηση της απόδοσης μειώνοντας ταυτόχρονα την κατανάλωση ενέργειας και χώρου. Για παράδειγμα, με βάση την προβλεπόμενη κίνηση, ο αριθμός των ενεργών εικονικών καναλιών μπορεί να ρυθμιστεί ώστε η κατανάλωση ενέργειας να μειωθεί.

5.4 Εφαρμογή GA σε NoC

Η εφαρμογή των γενετικών αλγορίθμων σε συστήματα NoC παρουσιάζει ιδιαίτερο ενδιαφέρον, αφού τα αποτελέσματα κυρίως όσον αφορά την ταχύτητα μεταφοράς δεδομένων και την κατανάλωση ενέργειας είναι σε ένα

πάρα πολύ καλό επίπεδο. Η λογική των γενετικών αλγορίθμων βασίζεται στο παρακάτω πρόβλημα [8]:

- Θεωρούμε ένα κατευθυνόμενο γράφο επικοινωνίας (CTG) G(V,E), όπου κάθε $v_i \in V$ δηλώνει είτε ένα στοιχείο επεξεργασίας, είτε μια μονάδα μνήμης (όπου θα την αναφέρουμε ως κόμβο) και κάθε ακμή $e_k = \{v_i, v_j\} \in E$ υποδηλώνει ένα ίχνος επικοινωνίας από v_i στο v_j. Για κάθε $v_i \in V$ το ύψος και το πλάτος του πυρήνα δηλώνετε με H_i και W_i αντίστοιχα. Ο CTG είναι μια γενικευμένη αναπαράσταση επικοινωνίας που επιτρέπει κύκλους και πολλαπλές ακμές ανάμεσα στους πυρήνες επεξεργασίας.

- Κάθε $e_k = \{v_i, v_j\} \in E, w(e_k)$, δηλώνει το εύρος ζώνης σε bits/cycle, και $\sigma(e_k)$ δηλώνει την καθυστέρηση σε hops.

- Στην αρχιτεκτονική του δρομολογητή, το η δηλώνει τον μέγιστο αριθμό εισόδων/εξόδων του συστήματος και το Ω δηλώνει το εύρος ζώνης εισόδου και εξόδου που μπορεί να υποστηριχτεί σε οποιαδήποτε θύρα του δρομολογητή. Έτσι κάθε θύρα ενός δρομολογητή μπορεί να υποστηρίξει ίσο εύρος ζώνης σε λειτουργία εισόδου ή εξόδου. Επειδή κάθε κόμβος $v \in V$ συνδέεται σε κάθε θύρα του δρομολογητή, το εύρος ζώνης σε κάθε κόμβο από ένα δρομολογητή και το εύρος ζώνης από ένα κόμβο σε κάθε δρομολογητή είναι πάντα μακρότερη του Ω. Δύο ποσότητες Ψ_i και Ψ_0 δηλώνουν την κατανάλωση ενέργειας ανά megabits/sec του εύρους ζώνης προς τις εισόδους ή τις εξόδους, για κάθε θύρα του δρομολογητή.

- Η ποσότητα $D_{\max}$ δηλώνει τη μέγιστη επιτρεπτή απόσταση που μπορεί να έχουν δύο δρομολογητές για να εξασφαλιστεί ενιαία μεταφορά δεδομένων σε κάθε κύκλο.

- Το φυσικό μοντέλο ισχύος σύνδεσης συμβολίζεται με Ψ_l και μετριέται σε nW per Mbps per sec.

- Σε επίπεδο συστήματος η κάτοψη των πυρήνων στο SoC:

Το R δηλώνει το σύνολο των δρομολογητών που συνθέτουν την αρχιτεκτονική μας, το E_r αντιπροσωπεύει το σύνολο των δεσμών μεταξύ δύο δρομολογητών και το E_v αντιπροσωπεύει το σύνολο των δεσμών μεταξύ των δρομολογητών και των κόμβων. Ο στόχος του σχεδιασμού ενός NoC είναι να δημιουργήσει μια τοπολογία δικτύου $T(R,V,E_r,E_v)$ τέτοια ώστε:

- Για κάθε $e_k = \{v_i, v_j\} \in E$ υπάρχει μια διαδρομή $p = \{(v_i, r_i), (r_i, r_j), ..., (r_k, v_j)\}$ με T τέτοιο ώστε να ικανοποιεί κάθε $\sigma(e_k)$ και $w(e_k)$.

- Να ικανοποιούνται οι περιορισμοί του εύρους ζώνης για τις θύρες των δρομολογητών.

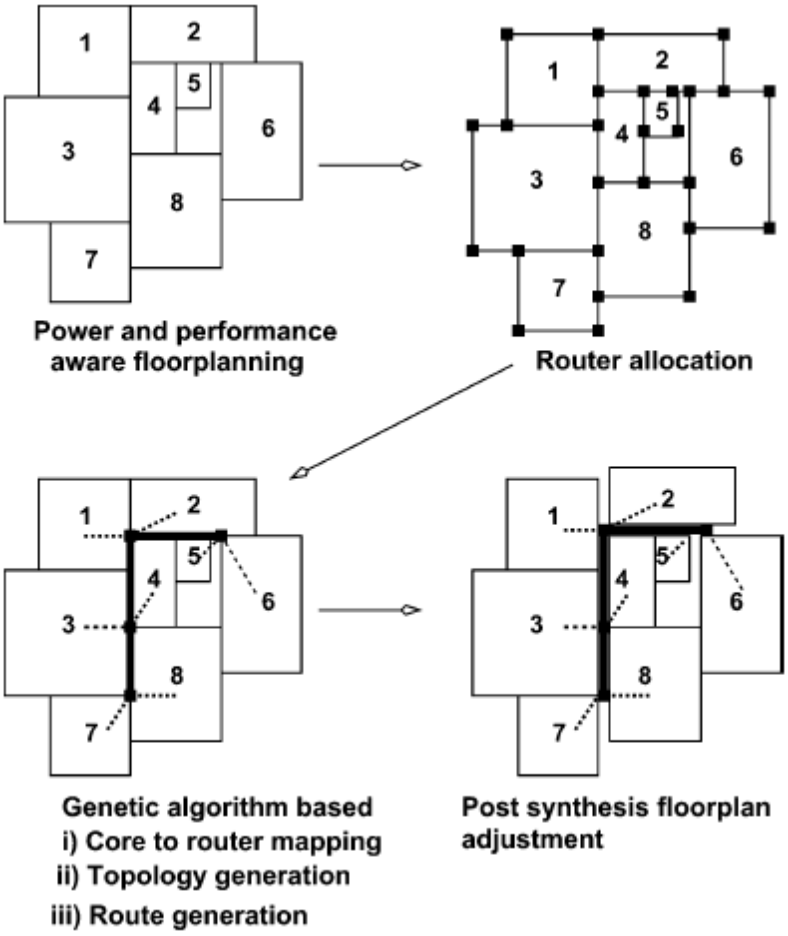
- Η συνολική κατανάλωση ενέργειας σε επίπεδο συστήματος για την inter-core επικοινωνία να ελαχιστοποιείτε.

Η εφαρμογή στο συγκεκριμένο πρόβλημα τοπολογίας αποτελεί παραλλαγή του γενικευμένου προβλήματος steiner forest [9], που είναι γνωστό ως NP hard. Μια έρευνα υφιστάμενων προσεγγίσεων που αφορά την εφαρμογή του προβλήματος πάνω σε αρχιτεκτονικές NoC έχει μελετηθεί από τον Benini [10]. Υπάρχουσες τεχνικές αρχιτεκτονικής NoC [11]-[16] έχουν αγνοηθεί σε μεγάλο βαθμό λόγω των περιορισμών που τίθενται είτε από το μήκος των φυσικών δεσμών, είτε στη κατανάλωση ενέργειας της διασύνδεσης των στοιχείων. Ο Srinivasan [12] παρατήρησε ότι οι φυσικές συνδέσεις στα συστήματα NoC αναμένετε να έχουν 30% μεγαλύτερη κατανάλωση ενέργειας. Σε αντίθεση με ντετερμινιστικές τεχνικές γραμμικού

προγραμματισμού [12], [17] στο συγκεκριμένο κεφάλαιο θα ασχοληθούμε με μια διαφορετική προσέγγιση βασισμένη σε γενετικούς αλγορίθμους δρομολόγησης (GA). Οι τεχνικές βασισμένες στους γενετικούς αλγορίθμους μας επιτρέπουν μια αποτελεσματικότερη προσέγγιση κυρίως όσον αφορά την μείωση της κατανάλωσης ενέργειας του συστήματος. Η τεχνική GA έχει τη δυνατότητα να δημιουργήσεις εξαιρετικής ποιότητας λύσεις σε εύλογο χρονικό διάστημα. Επιπροσθέτως η τεχνική GA μπορεί να δημιουργήσει ένα σύνολο σημείων Pareto όπου κάθε σημείο θα αντιπροσωπεύει μια λύση με μια συγκεκριμένη κατανάλωση ενέργειας και δρομολόγησης πόρων. Μας επιτρέπετε επίσης να έχουμε την δυνατότητα αλλαγής της επιθυμητής διαδρομής επιλέγοντας την επιθυμητή ανάμεσα στην βέλτιστη κατανάλωση ενέργειας και αυτής με τους μεγαλύτερους πόρους στο δρομολογητή μας.

5.5 Γενετικός Αλγόριθμος για τοπολογίες NoC

Η εφαρμογή της σχεδίασης ενός NoC χρησιμοποιώντας έναν GA απεικονίζεται στην εικόνα 5.4



Εικόνα 5.4 Ροή εφαρμογής για τη σύνθεση ενός NoC

Στην τεχνική αυτή παίρνουμε το γράφημα ενός ίχνους επικοινωνίας, μια κάτοψη σε επίπεδο συστήματος και τα στοιχεία της αρχιτεκτονικής διασύνδεσης ως είσοδο. Ως πρώτο βήμα η τεχνική αυτή τοποθετεί τους δρομολογητές σε φυσικές θέσεις. Η επιλογή της τοποθεσίας του κάθε δρομολογητή στο GA θα πρέπει να είναι τέτοια ώστε να μειωθεί σημαντικά ο χώρος και συνεπώς ο χρόνος εκτέλεσής της. Οι πιθανές θέσεις των δρομολογητών υποτίθεται ότι είναι στις γωνίες των πυρήνων, όπως απεικονίζεται στην εικόνα 5.4. Με δεδομένο ότι οι φυσικές θέσεις των δρομολογητών είναι γνωστές, μπορούμε να καθορίσουμε τις ελάχιστες αποστάσεις από κάθε κόμβο προς τους δρομολογητές. Με το ίδιο επιχείρημα,

μπορούμε επίσης να καθορίσουμε όλες τις αποστάσεις μεταξύ των δρομολογητών. Ως εκ τούτου μπορούμε να εκτιμήσουμε τα μήκη των συνδέσμων (και συνεπώς την κατανάλωση ενέργειας) με αρκετά μεγάλη ακρίβεια. Το φυσικό μήκος μεταξύ των κόμβων και μεταξύ κόμβων δρομολογητών περιορίζεται από τη συχνότητα του ρολογιού του επεξεργαστή μας. Ο σχεδιασμός μας μπορεί να καθορίσει και το μέγιστο φυσικό μήκος της ζεύξης που επιτρέπετε από την μέγιστη μεταφορά δεδομένων σε κάθε κύκλο.

Σαν επόμενο βήμα η τεχνική GA με αυτοματοποιημένη διαδικασία επιλέγει επιλέγει τους δρομολογητές που θα χρησιμοποιηθούν στο NoC, χαρτογραφεί τους κόμβους στους δρομολογητές, κατασκευάζει την τοπολογία δικτύου και δρομολογεί την κίνηση στην αρχιτεκτονική διασύνδεσης. Η τελική διάταξη της εικόνας 4 δείχνει με διακεκομμένες γραμμές τη διαδρομή από ένα κόμβο σε ένα δρομολογητή και με έντονη γραμμή την τοπολογία ενός NoC. Στο τελικό στάδιο, προσαρμόζουμε την κάτοψη του SoC με τέτοιο τρόπο ώστε η περιοχή των δρομολογητών να λαμβάνετε υπόψη. Ωστόσο, δεδομένου ότι οι διαστάσεις του δρομολογητή είναι πολύ μικρές δεν περιμένουμε σημαντική διακύμανση στις διαστάσεις του συστήματος στο τελικό στάδιο σε σχέση με τα προηγούμενα.

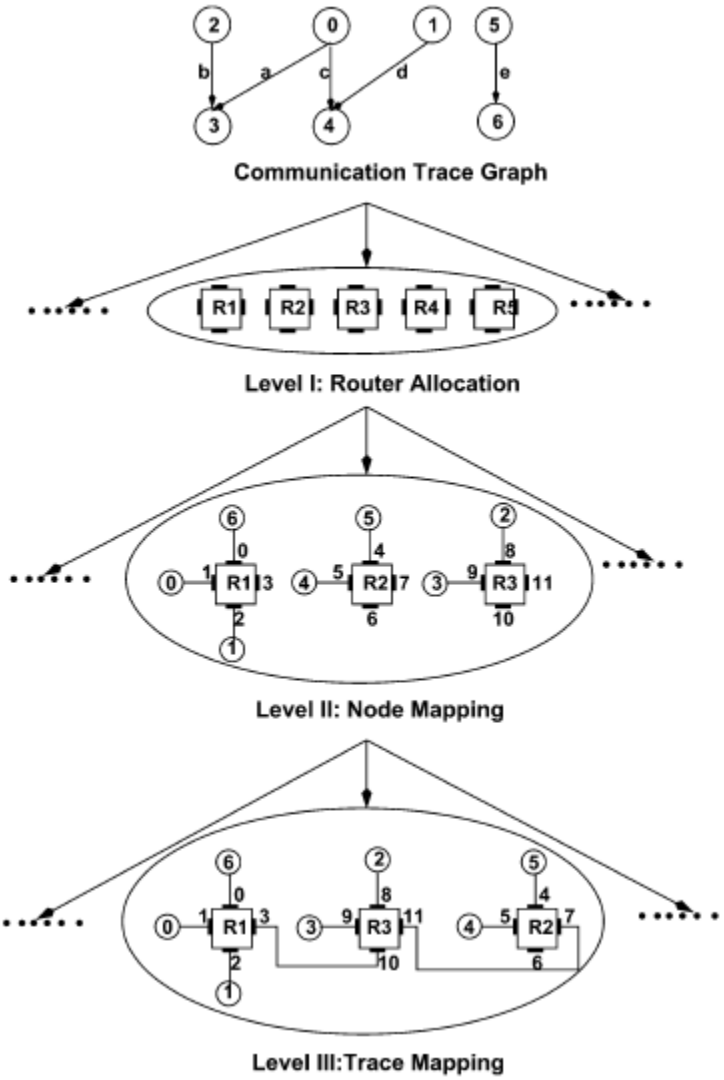
Οι αρχιτεκτονικές των NoC που δημιουργούνται με αυτή τη διαδικασία ενδέχεται να δημιουργήσει αδιέξοδα ανάμεσα στα trace routes. Ο στατικός αλγόριθμος δρομολόγησης που θα χρησιμοποιήσουμε εκμεταλλευόμενη την αρχιτεκτονική του NoC προσπαθεί να δημιουργήσει διαδρομές αποφεύγοντας τα αδιέξοδα. Σε περίπτωση που ανιχνευθούν αδιέξοδα σε κάθε βήμα επεξεργασίας προσθέτονται εικονικά κανάλια σε κατάλληλες θέσεις δρομολογητών για να σπάσουμε τα αδιέξοδα [1], [18].

5.5.1 Επισκόπηση GA

Ένας GA βασίζεται στο βιολογικό φαινόμενο της γενετικής εξέλιξης. Διατηρεί μια σειρά από λύσεις που είναι γνωστές ως generation. Λειτουργεί με επαναληπτικό τρόπο και εξελίσσεται συνεχώς από generation σε generation. Με τη δημιουργία μιας νέας generation, μετά την αύξηση του πληθυσμού, δημιουργεί νέες λύσεις και με κατάλληλα κριτήρια καταλληλότητας επιλέγει έναν σταθερό αριθμό από αυτές. Τα κριτήρια καταλληλότητας προκύπτουν από μια συνάρτηση που έχει στόχο την βελτιστοποίηση του συστήματός μας. Η επιλογή των λύσεων βασίζονται σε κριτήρια καταλληλότητας σε μοντέλα εξελικτικής συμπεριφοράς γνωστά και ως “survivor of fittest”. Ο αλγόριθμός μας λειτουργεί με επαναληπτικό τρόπο έως ότου επιτευχθεί το επιθυμητό αποτέλεσμα.

5.5.2 GA Data Structure

Η βελτιστοποίηση των GA-based αλγορίθμων βασίζεται στην αναπαραγωγή του πληθυσμού που υποστηρίζει την αποτελεσματική εφαρμογή γενετικών παραγόντων. Ο αλγόριθμος αυξάνει ιεραρχικά το πληθυσμό με μια τεχνική που αποτελείται από τρία επίπεδα. Το πρώτο επίπεδο αντιπροσωπεύει τον αριθμό των δρομολογητών, το δεύτερο υποδηλώνει την χαρτογράφηση των κόμβων CTG προς τις εισόδους των δρομολογητών και το τρίτο τη διαδρομή των πακέτων από την πηγή προς τους ενδιάμεσους δρομολογητές. Η εικόνα 5.5 μας δείχνει την ιεραρχική δομή ενός GA-based αλγορίθμου.

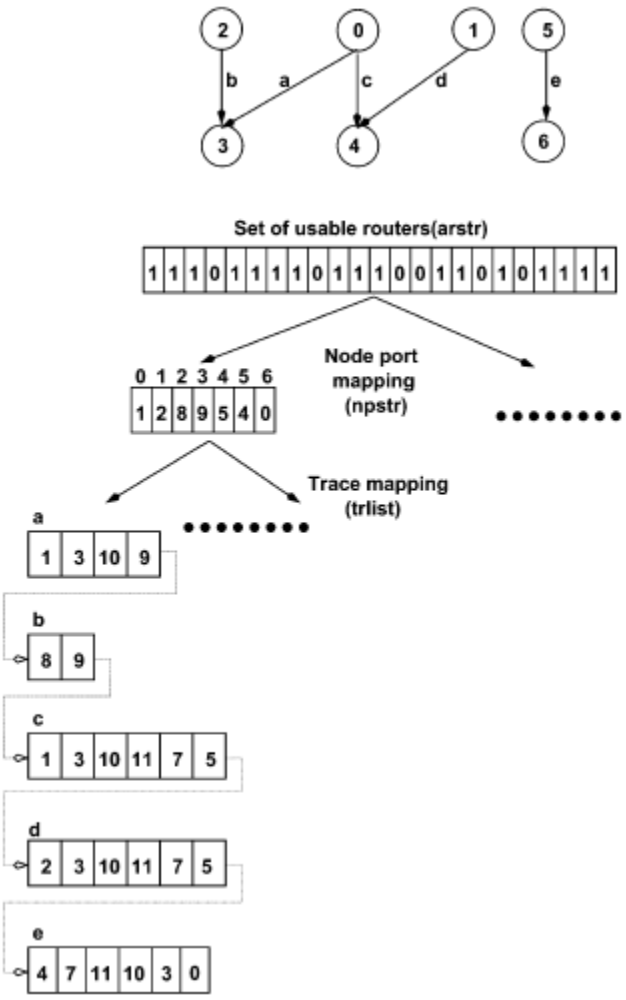


Εικόνα 5.5 Ιεραρχική δομή ενός GA-based αλγορίθμου

Στο πρώτο επίπεδο, η τεχνική μας διατηρεί σε κάθε αρχιτεκτονική διαφορετικούς αριθμούς δρομολογητών. Στο δεύτερο επίπεδο, για κάθε δρομολογητή του πρώτου επιπέδου, η τεχνική αποθηκεύει J διαφορετικούς κόμβους σε κάθε δρομολογητή. Ενώ στο τρίτο επίπεδο, η τεχνική διατηρεί K διαφορετικές χαρτογραφήσεις για κάθε θύρα των δρομολογητών προς κάθε κόμβο του συστήματος του επιπέδου 2. Αξίζει να επισημάνουμε ότι η τεχνική δεν μοντελοποιεί ρητά τις φυσικές συνδέσεις μεταξύ των δρομολογητών.

Αντίθετα οι φυσικές συνδέσεις προέρχονται από τη χαρτογράφηση των ιχνών επικοινωνίας.

1) Επίπεδο 1 (Επίπεδο Δομής Δρομολογητών): Η εφαρμογή των γενετικών αλγορίθμων απαιτεί την εκπροσώπηση του πληθυσμού των δρομολογητών ως χρωμοσώματα. Η εικόνα 5.6 απεικονίζει την αναπαράσταση της εικόνας 5.5 σε χρωμοσώματα.



Εικόνα 5.6 Αναπαράσταση δομής κόμβων σε πίνακα

Στο πρώτο επίπεδο, ο αριθμός δρομολογητών σε μία λύση αντιπροσωπεύεται από τον δυαδικό πίνακα arstr[MAXROUTER], όπου MAXROUTER ο συνολικός αριθμός δρομολογητών στην αρχιτεκτονική μας. Η θέση και ο αριθμός των δρομολογητών έχουν προσδιοριστεί από την εφαρμογή του GA. Ο MAXROUTER είναι ο μέγιστος αριθμός δρομολογητών που μπορεί να χρησιμοποιηθεί. Για το παράδειγμα των εικόνων 2.2 και 2.3 MAXROUTER=24. Συμβολίζουμε τον κάθε δρομολογητή με r_i, όπου i ένας ακέραιος για τον οποίο ισχύει $0 \le i \le MAXROUTER - 1$. Σε κάθε δρομολογητή ο οποίος θα μπορούσε να χρησιμοποιηθεί από την αρχιτεκτονική μας εκχωρείται μια τυχαία θέση στον πίνακα που δίνεται από το $loc(r_i)$. Για παράδειγμα στον r_4 μπορεί να ανατεθεί η τοποθεσία $arstr[0](loc(r_4) = 0)$, ενώ στον r_2 η τοποθεσία $arstr[1](loc(r_2) = 0)$ και ούτω καθ' εξής. Τελικά σε όλους τους δρομολογητές που ενδεχομένως να μπορούσαν να χρησιμοποιηθούν από την αρχιτεκτονική έχει ανατεθεί ένας μοναδικός αριθμός. Ως εκ τούτου στις τέσσερεις θύρες του r_4 στη θέση $arstr[0]$ δίνονται τα νούμερα 0, 1, 2, 3 στις τέσσερεις θύρες του r_2 στη θέση $arstr[1]$ οι αριθμοί 4, 5, 6, 7 … Ο GA διατηρεί I περιπτώσεις του πίνακα $arstr$ στο πρώτο επίπεδο. Στον δυαδικό πίνακα $arstr$ εκχωρείται η τιμή "1" για κάθε δρομολογητή ο οποίος θα μπορούσε να χρησιμοποιηθεί από την αρχιτεκτονική και ο αριθμός "0" για κάθε δρομολογητή ο οποίος δεν θα χρησιμοποιηθεί. Χρησιμοποιούμε τον όρο "θα μπορούσε να χρησιμοποιηθεί" διότι κάποιος δρομολογητής αν και συμπεριλαμβάνετε στην αρχιτεκτονική, θα μπορούσε οποιοδήποτε δρομολόγηση να μην περνά από αυτόν.

2) Επίπεδο 2 (Επίπεδο Μεταφοράς Δεδομένων): Όπως αναφέρθηκε για κάθε εμφάνιση του πίνακα, ο GA διατηρεί για όλες τις περιπτώσεις του πίνακα χαρτογράφηση των δρομολογητών. Ο κάθε κόμβος στο δεύτερο επίπεδο αποθηκεύεται σε ένα πίνακα ακέραιων τον $npstr$. Η κάθε τοποθεσία i έναν αριθμό θύρας για τον κάθε εκχωρημένο κόμβο

$v_i \in V$. Στην εικόνα 6 στον πίνακα ακεραίων *npstr* δίνεται η τιμή 0 στην θύρα που αντιστοιχεί στον χαρτογραφημένο κόμβο 1, η τιμή 1 στην θύρα που αντιστοιχεί στον χαρτογραφημένο κόμβο 2, η τιμή 1 στην θύρα που αντιστοιχεί στον χαρτογραφημένο κόμβο 8 και ούτω καθ' εξής. Από τη στιγμή που κάθε θύρα μπορεί να χαρτογραφηθεί μόνο σε ένα κόμβο, η θύρα δεν μπορεί να επαναληφθεί στη σειρά.

Η τεχνική μας μπορεί να δημιουργήσει τοπολογίες NoC με ετερογενείς δρομολογητές που έχουν μεταβλητό αριθμό εισόδων/εξόδων. Ο αριθμών των θυρών σε κάθε δρομολογητή γίνετε γνωστός μόνο μετά την επίτευξη της τελικής λύσης. Ως εκ τούτου η διαδικασία ξεκινά με ένα σταθερό αριθμό θυρών σε κάθε δρομολογητή. Για παράδειγμα, έστω ότι ο μέγιστος αριθμός θυρών σε ένα δρομολογητή είναι 8 (ο αριθμός αυτός προκύπτει από τη βιβλιοθήκη IP), GA αρχικά υποθέτει ότι όλοι οι δρομολογητές έχουν 8 θύρες. Μόλις δημιουργηθεί η τοπολογία ο απαιτούμενος αριθμός θυρών σε κάθε δρομολογητή είναι γνωστός και το κατάλληλο IP-block μπορεί να επιλεχθεί κατάλληλα από τη βιβλιοθήκη. Έτσι, η τοπολογία το NoC μας υποστηρίζει ετερογενείς τοπολογίες δρομολογητών.

3) Επίπεδο 3 (Επίπεδο δομής δεδομένων): Ο GA διατηρεί K περιπτώσεις μεταφοράς πακέτων για κάθε θέση του πίνακα *npstr* . Η χαρτογράφηση ενός ίχνους επικοινωνίας δηλώνετε σε μια λίστα *trlist* ακέραιων πινάκων $trstr_i$ όπου το i αναφέρεται στην i χαρτογραφημένη επικοινωνία ίχνους. Για παράδειγμα στην εικόνα 5.3, ο πρώτος πίνακας *srstr* αντιστοιχίζεται στη διαδρομή a, ο δεύτερος στη διαδρομή b και ούτω καθ' εξής. Κάθε *srstr* πίνακας είναι ένα διατεταγμένο σύνολο δρομολογητών που μας δείχνει τη ροή των δεδομένων. Για παράδειγμα το ίχνος επικοινωνίας a ανάμεσα στους δρομολογητές 1, 3, 10, 9 αντιπροσωπεύεται από μια συστοιχία που δίνεται από τον πίνακα $[1, 3, 10, 9]$.

5.5.3 Κριτήρια επίλυσης διαδρομών GA

Σε αυτή την ενότητα, θα παρουσιάσουμε τα αναγκαία και επαρκή κριτήρια που πρέπει να ισχύουν ώστε να έχουμε την πλέον ενδεδειγμένη λύση σε επίπεδο δρομολογητών, κόμβων και διαδρομών. Η παραβίαση των κριτηρίων αυτών θα μας οδηγήσει σε ανέφικτες λύσεις.

Αξίζει να επισημάνουμε ότι κάθε κόμβος αντιστοιχίζεται σε ένα μόνο δρομολογητή. Ως εκ τούτου, σε επίπεδο router, το μόνο κριτήριο που θα πρέπει να πληρούται είναι ότι ο συνολικός αριθμός ports κάθε δρομολογητή θα πρέπει να είναι μεγαλύτερος ή ίσος με τον αριθμό των κόμβων στο αντίστοιχο γράφημα CTG.

Στο επίπεδο κόμβων θα πρέπει να ισχύουν τα παρακάτω κριτήρια.

1) Από τη στιγμή όπου κάθε θύρα μπορεί να χαρτογραφηθεί σε ένα μόνο κόμβο, ο κόμβος μπορεί να χαρτογραφηθεί μόνο σε μια θέση του πίνακα .

2) Μια θύρα δεν μπορεί να ανήκει σε ένα δρομολογητή ο οποίος δεν περιλαμβάνετε στο συγκεκριμένο επίπεδο. Ως εκ τούτου όλες οι θύρες που έχουν χαρτογραφηθεί στον πίνακα $srstr$ ανήκουν σε δρομολογητή r_i ο οποίος περιλαμβάνετε στο επίπεδο δρομολογητών των υποψήφιων διαδρομών, ο οποίος είναι ο $arstr\left[loc\left(r_i\right)\right]=1$.

3) Η απόσταση μεταξύ κόμβου και θύρας πρέπει να είναι μικρότερη από την μέγιστη προκαθορισμένη για καταστεί δυνατή η μεταφορά δεδομένων σε κάθε κύκλο του ρολογιού.

Ο GA θα πρέπει να ελέγξει αρκετές συνθήκες νομιμότητας σε επίπεδο ίχνους οι οποίες συνοψίζονται ως εξής:

1) Κάθε ίχνος κυκλοφορίας αντιστοιχίζεται σε ένα πίνακα.

2) Στον πίνακα στη θέση 0 τοποθετείται η θύρα εισόδου του κάθε δρομολογητή. Ακολουθούν οι υπόλοιπες θύρες του δρομολογητή και ο πίνακάς κλείνει με έναν ακέραιο που αντιπροσωπεύει την αντίστοιχη θύρα εξόδου.

3) Κάθε θύρα που εκχωρείται σε κάποιο κόμβο αντιπροσωπεύει είτε την αφετηρία είτε τον τερματισμό της κάθε πιθανής διαδρομής του κάθε κόμβου. Ως εκ τούτου καμία άλλη θύρα του πίνακα δεν αντιστοιχίζεται στον εκάστοτε κόμβο.

4) Αν ένα ίχνος εισέλθει σε κάποιο δρομολογητή, θα πρέπει να εξέλθει από κάποια θύρα εξόδου. Προκειμένου να ξεπεραστεί το συγκεκριμένο εμπόδιο υιοθετούμε τη σύμβαση ότι οι άρτιες θέσεις του πίνακα αντιπροσωπεύουν θύρες στις οποίες εισέρχεται ένα ίχνος και οι περιττές τις θύρες τις οποίες εξέρχεται.

5) Με βάση τον προηγούμενο περιορισμό, αν ένα ίχνος εισέρχεται σε μια θύρα, η επόμενη θύρα θα είναι θύρα εξόδου.

6) Για να αποφευχθούν τυχόν κύκλοι στη δρομολόγησή μας, ο αριθμός κάθε θύρας δεν επαναλαμβάνετε.

7) Δεδομένου ότι η διαδρομή δεν περιέχει κύκλους, το πολύ δύο θύρες ενός δρομολογητή θα περιέχονται στη διαδρομή μας. Παραβίαση αυτού του κανόνα συνεπάγετε και την πιθανότητα δημιουργίας κύκλων στη διαδρομή μας. Μαζί με τους περιορισμούς 4, 5 συμπεραίνουμε ότι σε κάθε κόμβο οι χρησιμοποιούμενες θύρες θα είναι είτε μηδέν είτε δύο.

8) Κάθε ζευγάρι θυρών των δρομολογητών μας θα πρέπει να συστοιχιστεί με τέτοιο τρόπο στον πίνακά μας, ώστε μεταξύ των διαφορετικών δρομολογητών να υπάρχει φυσική σύνδεση μεταξύ τους. Αυτός ο περιορισμός εξασφαλίζει ότι κάθε θύρα θα συνδεθεί με μία και μόνο θύρα παρακείμενου δρομολογητή.

9) Αν ένα πακέτο μετακινηθεί μέσω ενός δρομολογητή, τότε υποχρεωτικά θα περάσει από δύο θύρες του εν λόγο δρομολογητή. Σε συνδυασμό αυτές οι δύο θύρες καθιστούν ένα hop. Ως εκ τούτου το μήκος του θα πρέπει να είναι ίσο ή μικρότερο με το διπλάσιο του μήκους της λανθάνουσας κατάστασης. Όπου λανθάνουσα κατάσταση είναι ο αριθμός που αντιπροσωπεύεται με τον συνολικό αριθμό hops.

10) Η απόσταση μεταξύ δύο παρακείμενων δρομολογητών στον πίνακα που αντιπροσωπεύει τα ίχνη πρέπει να επιτρέπει τη μεταφορά μιας λέξης σε κάθε κύκλο του ρολογιού. Όπως έχει προαναφερθεί έχει ήδη υπολογιστεί η μέγιστη απόσταση μεταξύ δύο δρομολογητών, ώστε να καταστεί δυνατό κάτι τέτοιο. Έτσι ο περιορισμός αυτός μπορεί να ικανοποιηθεί αν το φυσικό μήκος μεταξύ δύο δρομολογητών είναι μικρότερος από την καθορισμένη απόσταση που έχει καθορίσει ο σχεδιαστής.

11) Οι περιορισμοί στο εύρος ζώνης των θυρών του πίνακα δεν παραβιάζονται.

5.5.4 Δημιουργία Αρχικού πληθυσμού και Modified Shortest Path Algorithm (MSP)

Ο αρχικός μας πληθυσμός οριοθετείται δημιουργώντας I πίνακες δρομολογητών J πίνακες αντιστοιχίας κόμβων σε θύρες για κάθε κατανομή των δρομολογητών και K συστοιχίες χαρτογράφησης ιχνών επικοινωνίας για κάθε χαρτογράφηση κόμβων. Η αρχική κατανομή του δρομολογητή δημιουργείται από τυχαία εκχώρηση των “0” και “1” σε όλες τις θέσεις του πίνακα $arstr$. Ομοίως, η χαρτογράφηση κάθε κόμβου ως προς τις θύρες επιτυγχάνεται με ομοιόμορφη τυχαία επιλογή ενός κόμβου του CTG και χαρτογράφησή του σε μία θύρα η οποία επίσης επιλέγεται από μια ομοιόμορφα τυχαία λειτουργία.

Αρχικά, επιλέγουμε τυχαία ένα ίχνος και εφαρμόζουμε ένα MSP για να δημιουργήσουμε τη χαρτογράφηση της κίνησης από τον κόμβο πηγή στον κόμβο προορισμού. Όπως προαναφέρθηκε ή χαρτογράφηση ενός ίχνους δηλώνεται σε ένα πίνακα. Κάθε ίχνος που χαρτογραφείται δημιουργεί και μια φυσική σύνδεση μεταξύ δύο διαδοχικών θυρών των δρομολογητών μας. Ο αλγόριθμος για το MSP διαφέρει από τον κλασσικό shortest path αφού χαρτογραφεί την κίνηση κατά μήκος της συντομότερης διαδρομής που αντιστοιχούν στους συνδέσμους που έχουν δημιουργηθεί από τα ήδη χαρτογραφημένα ίχνη και τους περιορισμούς του εύρους ζώνης στις θύρες των δρομολογητών. Επιπλέον οι διαδρομές που δημιουργούνται είναι τέτοιες ώστε η πιθανότητα deadlock να ελαχιστοποιείται.

Δεδομένου ότι ο αλγόριθμος MSP είναι ικανός να εντοπίσει διαδρομές με ελάχιστη κατανάλωση ενέργειας, η απόσταση μεταξύ δύο οποιωνδήποτε δρομολογητών μετράτε ως προς την κατανάλωση ρεύματος. Για δύο δρομολογητές r_1 και r_2 το κόστος εγκατάστασης της διαδρομής για ένα ίχνος t δίνεται από τον τύπο:

$$c_{(r_1,r_2)} = \begin{cases} \left(\Psi_i + \Psi_0 + dist\left(r_1, r_2\right) \times \Psi_t\right) \times \omega(t), \alpha\nu\, dist\left(r_1, r_2\right) \le D_{\max} \kappa\alpha\iota\, \delta\varepsilon\nu\, \acute{\varepsilon}\chi ουμε \\ \pi\alpha\rho\alpha\beta\acute{\iota}\alpha\sigma\eta\, \kappa\alpha\nu\varepsilon\nu\acute{ο}ς\, \pi\varepsilon\rho\iota ορισμού\, \mu\varepsilon\tau\alpha\xi\acute{ύ}\, \tau\omega\nu\, \delta ρομολογητών\, r_1\, \kappa\alpha\iota\, r_2 \\ \infty, \sigma\varepsilon\, ο\pi οιαδήποτε\, \acute{\alpha}\lambda\lambda\eta\, \pi\varepsilon\rho\acute{\iota}\pi τωση \end{cases}$$

Όπου $dist\left(r_1, r_2\right)$ η απόσταση Manhattan μεταξύ των δρομολογητών r_1 και r_2 και $D_{\max}$ η μέγιστη επιτρεπόμενη απόσταση μεταξύ δύο δρομολογητών ώστε να εξασφαλιστεί η μεταφορά των δεδομένων σε έναν κύκλο ρολογιού. Ωστόσο η συνάρτηση κόστους καθορίζει τις ακόλουθες δύο ιδιότητες.

- Η συντομότερη διαδρομή αντιστοιχεί και στη διαδρομή με τη μικρότερη κατανάλωση ενέργειας.

- Αν οι δρομολογητές r_1 και r_2 έχουν απόσταση μεγαλύτερη της επιτρεπτής ή το bandwidth είναι μεγαλύτερο από αυτό που μπορούν να εξυπηρετήσουν οι δύο δρομολογητές μας τότε το κόστος τίθεται ίσο με το ∞ και η σύνδεση μεταξύ των συγκεκριμένων δυο δρομολογητών αποκόπτεται από το σύστημά μας.

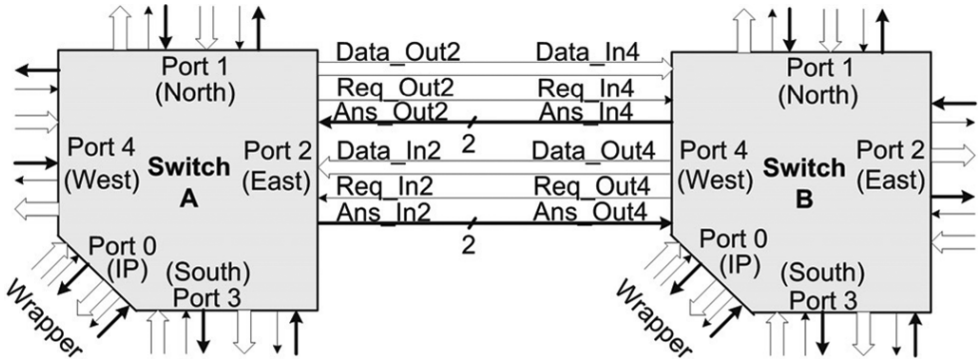
5.6 Backtracking switching

Η τοπολογία torus είναι η πλέον ενδεδειγμένη για backtracking τόσο σε 2-D noc όσο και για 3-D. Η απ' άκρο σε άκρο επικοινωνία επιτυγχάνεται αφού σχεδιαστούν και εξεταστούν λεπτομερώς τα μονοπάτια επικοινωνίας.

5.6.1 Λειτουργία ελέγχου ροής από άκρο σε άκρο με backtracking switching

Μια τυπική επικοινωνία από άκρο σε άκρο όπως χρησιμοποιείται στην προσέγγιση μεταγωγής κυκλώματος μέσω pipeline παρέχει μια λεπτομερή περιγραφή της backtracking τεχνικής στους μεταγωγείς του κυκλώματός μας. Η επικοινωνία περιλαμβάνει τρεις βασικές φάσεις: την διαμόρφωση της διαδρομής (ή την ανίχνευση), τη μετάδοση και την εξαγωγή. Στη φάση ρύθμισης της διαδρομής η κεφαλίδα που περιέχει τη διεύθυνση προορισμού στέλνεται από την πηγή στον προορισμό για να ρυθμιστεί ένα κύκλωμα επικοινωνίας. Όταν υπάρξει συμφόρηση στο σύστημά μας, το σύστημα αναζητά εναλλακτικούς συνδέσμους, αντί κάποιοι από τους δεσμούς του συστήματος να παραμένουν αδρανείς. Δηλαδή η διαδρομή έχει ρυθμιστεί με σύστημα backtracking. Όταν το σήμα μας φτάσει στον προορισμός μας τότε στέλνεται στην πηγή ένα σήμα ACK. Στην συνέχεια, στη φάση της μετάδοσης, η πηγή αρχίζει να μεταδίδει δεδομένα συγχρονισμένα με την

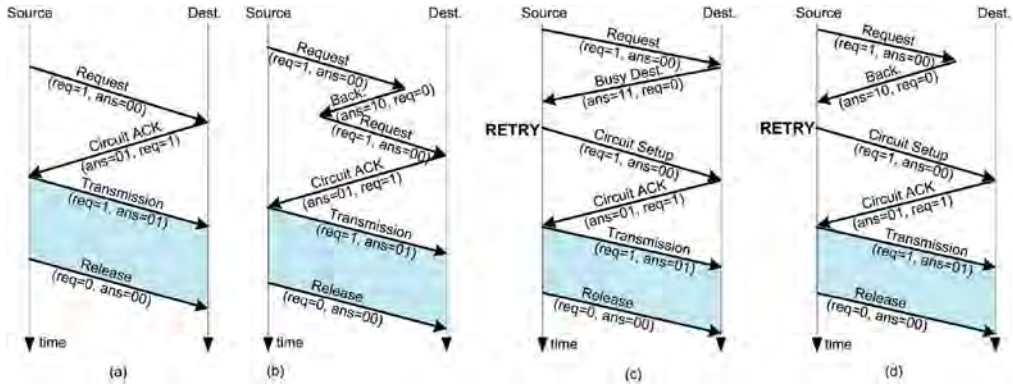
πηγή μέσω της ρυθμισμένης διαδρομής προς τον προορισμό. Το ρολόι είναι χρονισμένο με τα δεδομένα και χρησιμοποιείται για να διατηρεί το χρονοδιάγραμμα χρονισμού των ρολογιών του συστήματος. Στη φάση της εξαγωγής, το κύκλωμα απελευθερώνεται με τρόπο hop-by-hop από την πηγή ως τον προορισμό.



Εικόνα 5.7 Switch-by-Switch διασύνδεση

Στην εικόνα 5.7 απεικονίζεται η διασύνδεση μεταξύ των πυρήνων του συστήματός μας. Κάθε πυρήνας έχει πέντε αμφίδρομες θύρες. Οι τέσσερεις συνδέονται με τους γειτονικούς πυρήνες του συστήματος και η πέμπτη μέσω wrapper συνδέεται με το chip μας.

Σύμφωνα με το σχέδιο “χειραψίας” μεταξύ των πυρήνων, χρησιμοποιείται ένα bit για το σήμα Request (Req) για να δηλωθεί κατάσταση ανίχνευσης σήματος και να δηλωθεί η κατάσταση ηρεμίας του συστήματος. Δύο bits χρησιμοποιούνται για το σήμα Answer (Ans). Αυτό περιέχει μια από τις τρεις περιπτώσεις. Κατάσταση ‘01’ που δηλώνει ότι ο δέκτης είναι έτοιμος να δεχθεί δεδομένα από τον αποστολέα, την κατάσταση ‘10’ που δηλώνει ότι η προβλεπόμενη διαδρομή είναι αποκλεισμένη αναγκάζοντας το σύστημα να ανακαλύψει νέες πιθανές διαδρομές και την κατάσταση ‘11’ που δηλώνει ότι ο δέκτης δεν είναι έτοιμος να δεχθεί δεδομένα (π.χ. λόγω του ότι είναι απασχολημένος ή έχει γεμίσει το buffer του).



Εικόνα 5.8 Παραδείγματα της λειτουργίας ελέγχου ροής από άκρο σε άκρο που χρησιμοποιεί χειραψία switch-by-switch α) επιτυχής ρύθμιση διαδρομής χωρίς backtracking β) επιτυχής ρύθμιση διαδρομής με backtracking γ) αποτυχία ρύθμισης διαδρομής λόγω κατειλημμένου προορισμού και επανάληψη δ) Αποτυχημένη ρύθμιση διαδρομής αφού όλες οι δυνατές διαδρομές είναι μπλοκαρισμένες και επανάληψη προσπάθειας σύνδεσης

Τα διαγράμματα χειραψίας switch-by-switch με pipeline στην από άκρο σε άκρο επικοινωνία απεικονίζονται στην παραπάνω εικόνα. Η εικόνα 5.8.α απεικονίζει ένα παράδειγμα επικοινωνίας από άκρο σε άκρο όπου το μονοπάτι επικοινωνίας έχει ρυθμιστεί σωστά χωρίς αναστροφή. Κατά τη διάρκεια της ρύθμισης φάσεων και μετάδοσης το Req έχει ρυθμιστεί σε ‘1’. Όταν το Ans είναι ‘00’ το πακέτο μας μετακινείται μέχρι να φτάσει στον προορισμό. Τότε ο προορισμός επιστρέφει Ans = ‘01’στην πηγή μας. Όταν η πηγή μας το λάβει ξεκινά να διαβιβάζει τα δεδομένα. Στη συνέχεια η πηγή μας θέτει το Req σε ‘0’ για να ελευθερώσει το κύκλωμα μετά την αποστολή των δεδομένων. Το παράδειγμα της εικόνας 5.8.β είναι παρεμφερές με αυτό της εικόνας 5.8.α αλλά αυτή τη φορά κατά η φάση της εγκατάστασης επικοινωνίας πραγματοποιείται με backtracking. Σε αυτό το παράδειγμα λόγω ενός μπλοκαρισμένου συνδέσμου ένας ενδιάμεσος διακόπτης ανατροφοδοτεί το Ans = ‘10’ στον προηγούμενο διακόπτη δημιουργώντας το backtracking.

Στην εικόνα 5.8.γ και 5.8.δ απεικονίζει τη χρήση σήματος Ans όταν αποτυγχάνεται η εγκατάσταση μιας διαδρομής και στη συνέχεια μια επανάληψη επαναλαμβάνεται. Στο παράδειγμα της εικόνας 5.3.γ όταν μια κεφαλίδα φτάσει σε έναν απασχολημένο προορισμό το σήμα Ans = ‘11’ στέλνεται προς τα πίσω στη πηγή μας και οι κατειλημμένες συνδέσεις ελευθερώνονται αντίστοιχα. Στο παράδειγμα της εικόνας 5.8.δ δεν υπάρχει διαθέσιμη διαδρομή που να οδηγεί στον προορισμό, οπότε στέλνετε απάντηση Ans = ‘10’ (δηλ. το δίκτυο είναι μπλοκαρισμένο). Σε ένα δυναμικό σενάριο αυτή η απάντηση δηλώνει την κατάσταση δικτύου (δηλ. μπλοκαρισμένο) στον αποστολέα. Στις περιπτώσεις όπου το Ans = ‘11’ (Busy Des.) και Ans = ‘10’ (Network Blocked) η πηγή υπό τον έλεγχο ενός πρωτοκόλου υψηλού επιπέδου επαναλαμβάνει να επαναλάβει την διαδικασία.

Βιβλιογραφία

Βιβλιογραφία

[1] W. J. Dally and B. Towles, “Route packet, not wires: On-chip interconnection networks,” in *Proc. DAC*, Jun. 2002, pp. 684–689.

[2] L. Benini and G. De Micheli, “Networks on chips: A new SoC paradigm,” *IEEE Computer*, vol. 35, no. 1, pp. 70–78, Jan. 2002.

[3] ST Microelectronics, Geneva, Switzerland, “ST network on chip,” 2005 [Online]. Available: http://www.st.com/stonline/press/news/back2005/b9014t.htm

[4] M. Melanie (1999), “An Introduction to Genetic Algorithms”, Cambridge, MA: MIT Press. ISBN 9780585030944.

[5] Tom, v.M n.d., “Genetic Algorithm”, http://www.civil.iitb.ac.in/tvm/2701_dga/

2701-ga-notes/gadoc/

[6] E. Schultz, J. Mellander & C. Endorf (2008), “Intrusion Detection & Prevention - A basic genetic algorithm”, [image online], http://my.opera.com/blu3c4t/blog/

show.dml/2636486.

[7] M. Bakhouya. Towards a Bio-Inspired Architecture for Autonomic Networkon-

Chip. In *Proceedings of the Workshop on Autonomic and High Performance Computing (AHPC’10)*, pages 491–497, 2010.

[8] K. Srinivasan, K. S. Chatha, and G. Konjevod, “Linear programming based techniques for synthesis of network-on-chip architectures,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 4, pp. 407–420, Apr. 2006.

[9] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt, III, “Approximation algorithms for degree-constrained minimum-cost network-design problems,” *Algorithmica*, vol. 31, no. 1, pp. 58–78, 2001.

[10] L. Benini, “Application specific NoC design,” in *Proc. DATE*, Mar. 2006, pp. 491–495.

[11] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli, “Efficient synthesis of networks on chip,” in *Proc. ICCD*, 2003, pp. 146–150.

[12] A. Jalabert, S. Murali, L. Benini, and G. De Micheli, “Xpipescompiler: A tool for instantiating application specific networks on chip,” in *Proc. DATE*, 2004, pp. 884–889.

[13] K. Srinivasan, K. S. Chatha, and G. Konjevod, “Linear programming based techniques for synthesis of network-on-chip architectures,” presented at the ICCD, San Jose, CA, Oct. 2004.

[14] K. Srinivasan and K. S. Chatha, “ISIS: A genetic algorithm based technique for synthesis of on-chip interconnection networks,” presented at the VLSI Des., Calcutta, India, Jan. 2005.

[15] U. Ogras and R. Marculescu, “Energy and performance driven NoC communication architecture synthesis using a decomposition approach,” in *Proc. DATE*, 2005, pp. 352–357.

[16] U. Ogras and R. Marculescu, “Application specific network-on-chip architecture customization via long range link insertion,” in *Proc. ICCAD*, 2005, pp. 246–253.

[17] K. Srinivasan and K. S. Chatha, “A technique for design of application specific network-on-chip architectures,” presented at the DATE, Munich, Germany, Mar. 2006.

[18] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks: An Engineering Approach*. Los Alamitos, CA: IEEE Computer Society, 1997.

$6^{ο}$ ΚΕΦΑΛΑΙΟ

ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

6.1 Εισαγωγή

Η δρομολόγηση στα noc είναι αρκετά παρόμοια με αυτή των κλασικών δικτύων. Όλοι οι αλγόριθμοι που αναπτύχθηκαν στα προηγούμενα κεφάλαια καθορίζουν τον τρόπο μεταφοράς των δεδομένων από τον αποστολέα στον παραλήπτη.

Σε γενικές γραμμές θα λέγαμε ότι οι αλγόριθμοι δρομολόγησης στα συστήματα noc χωρίζονται σε δύο μεγάλες κατηγορίες τους προσαρμοστικούς και τους μη-προσαρμοστικούς. Οι μη-προσαρμοστικοί χωρίζονται σε δύο μεγάλες υπο-ομάδες τους ντετερμινιστικούς και τους στοχαστικούς. Τα πακέτα της συγκεκριμένης κατηγορίας αλγορίθμων δεν περιέχουν πληροφορίες στην κεφαλίδα τους σχετικά με τη συμφόρηση και την κατάσταση του δικτύου. Στους προσαρμοστικούς αλγορίθμους τα πακέτα δεν ακολουθούν όλα την ίδια διαδρομή προσαρμόζοντας την δρομολόγηση τους ανάλογα με τις επικρατούσες συνθήκες του δικτύου.

Για τις γενικές κατηγορίες αλγορίθμων δρομολόγησης στα συστήματα noc οι διάφορες τεχνικές που μπορούν να χρησιμοποιηθούν συνοψίζονται στους παρακάτω πίνακες.

ALGORITHM	OUTLINES	FEATURES
Dimension order [1]	routing in one dimension at a time	simple
XY [2]	routing first in X and then in Y dimension	simple, loads network deadlock- and livelock-free
Pseudo adaptive XY [2]	partly adaptive XY routing	livelock-free, congestion avoidance
Surrounding XY [3]	partly adaptive XY routing	congestion avoidance
Turn model [20]	some turns forbidden	livelock-free
Valiant's random [1]	partly stochastic	balances network's load
Source [4, 5]	deterministic, sender determines the route	simple routing
Destination-tag [1, 6, 7]	deterministic, routers determine the route	simple sending
ALOAS [8]	deterministic, application of source routing	fast routing
Topology adaptive [9, 10]	Reprogrammable routing tables	suitable to dynamic networks
Probabilistic flood [11]	stochastic	cheap, consumes a lot of resources
Directed flood [11]	stochastic	fault-tolerant, consumes a lot of resources
Random walk [11]	stochastic	fault-tolerant

Πίνακας 1 Oblivious routing algorithms

ALGORITHM	OUTLINES	FEATURES
Minimal adaptive [1]	shortest path routing	simple
Fully adaptive [1]	congestion avoidance	non-minimal
Congestion look ahead [12]	congestion avoidance	fast
Turnaround / Turnback [13, 14, 15]	routing in butterfly - and tree networks	uses shortest path
Turn Back When Possible [16]	routing in tree networks	uses efficiently whole network
IVAL [16]	improved turnaround routing	uses efficiently whole network
2TURN [16]	slightly determined	efficient
Q [17]	statistics based routing	uses the best path
Odd-Even [18]	turn model	deadlock free
Slack-time aware [19]	routing for real-time applications	uses network resources efficiently
Hot-potato [20]	routing without buffer memories	cheap, sometimes misrouting

Πίνακας 2 Adaptive routing algorithms

Ενώ οι αλγόριθμοι δρομολόγησης των noc οι οποίοι βασίζονται κυρίως στην αρχιτεκτονική που έχει ο δρομολογητής μας παρουσιάζονται στον παρακάτω πίνακα

ROUTER	TOPOLOGY	FLOW CTRL	ALGORITHM	SPECIAL
Oblivious				
VCR [21]	2-dimensional	Wormhole	Source routing	Virtual channels
Xpipes [22]	Any	Wormhole	Source routing	Well adaptable
Æthereal [23]	Mesh	Wormhole	Contention free source routing	Combined GT and BE
Proteo [24]	Ring and subnets	Wormhole	Destination-tag	Layered structure
MANGO [25]	Mesh	Wormhole	Source routing	GT and BE traffic
SoCBUS [26]	Mesh	Store-and-forward	Destination-tag	Circuit switching
Arteris [27]	User-defined	User-defined	User-defined	Commercial
STNoC [28]	Spidergon	Wormhole	Source routing	Commercial
Adaptive				
DyAD [29]	Mesh	Wormhole	XY, Odd-Even	Dynamically deterministic
SPIN [13]	Fat tree	Wormhole	Turn around	Fault-tolerant
XGFT [14]	Fat tree	Wormhole variant	Turn around, source routing	Fault-tolerant
Nostrum [30]	Mesh	Virtual cut-through	Hot-potato	No buffers

Πίνακας 3 Router architectures

6.2 Συμπεράσματα

Το Network on Chip είναι μια τεχνολογία του μέλλοντος σε εφαρμογές System on Chip. Η τεχνολογία NoC είναι σχετικά μικρή σε μέγεθος και ως τώρα έχει ελάχιστες υλοποιήσεις. Υπάρχουν πολύ λίγες εμπορικές εφαρμογές του Network on Chip μέχρι στιγμής. Ωστόσο, αναμένεται ότι το NoC θα είναι μια κοινή τεχνολογία στο μέλλον. Το μικρό μέγεθος των κυκλωμάτων Network on Chip θέτει όλες τις ειδικές απαιτήσεις για όλες τις λειτουργίες. Η τεχνολογία δικτύου του Διαδικτύου είναι πολύ δύσκολο να συρρικνωθεί

ευθέως σε ένα NoC, οπότε οι τεχνολογίες πρέπει να προσαρμοστούν ειδικά στο NoC. Οι αλγόριθμοι δρομολόγησης που παρουσιάζονται στην παρούσα εργασία είναι δύσκολο να καθοριστούν με σειρά ανωτερότητας. Οι διαφορετικές εφαρμογές χρειάζονται διαφορετικούς αλγορίθμους δρομολόγησης. Ενώ κάποιος αλγόριθμος είναι κατάλληλος για ένα σύστημα, ένας άλλος αλγόριθμος λειτουργεί καλύτερα σε κάποιο άλλο σύστημα. Ωστόσο, μπορεί να γενικευθεί ότι στις περισσότερες περιπτώσεις ένας απλός αλγόριθμος ταιριάζει σε απλά συστήματα ενώ σύνθετοι αλγόριθμοι ταιριάζουν σε πιο πολύπλοκα συστήματα. Τα μεγάλα ποσά κυκλοφορίας δικτύου σε πολύπλοκα συστήματα απαιτούν αποτελεσματική λειτουργία κυκλοφορίας και αποφυγή συμφόρησης, ενώ τα πιο σημαντικά χαρακτηριστικά στα μικρότερα συστήματα είναι η χαμηλή κατανάλωση ενέργειας και η χαμηλή λανθάνουσα κατάσταση.

Σχεδόν όλες οι προτεινόμενες εφαρμογές Network on Chip είναι μεταγωγείς πακέτων και χρησιμοποιούν έλεγχο ροής δικτύων σκουληκότρυπας, η οποία είναι συνέπεια χαμηλότερων λανθάνων και μικρότερων αναγκών ρυθμιστικών μνημών σε αντίθεση με άλλες μεθόδους ελέγχου ροής. Ο πιο συνηθισμένος αλγόριθμος δρομολόγησης είναι η καθοριζόμενη δρομολόγηση πηγής. Ακόμα υπάρχουν προτεινόμενες υλοποιήσεις με τη χρήση προσδιοριστικής δρομολόγησης ετικετών προορισμού και προσαρμοστικών αλγορίθμων, όπως η περιστροφή και η δρομολόγηση των ζεστών γεωμήλων. Επιπλέον, οι πιο δημοφιλείς τοπολογίες δικτύου είναι το πλέγμα και το λίπος. Ο αριθμός των εφαρμογών των άλλων τοπολογιών είναι πολύ μικρός.

Οι περισσότερες από τις προτεινόμενες αρχιτεκτονικές δρομολογητών είναι ντετερμινιστικές. Όταν οι διαστάσεις των συστημάτων μειώνονται και τα συστήματα αναπτύσσονται προς την κατεύθυνση της νανοκλίμακας, η ανάγκη για συστήματα ανθεκτικών σε σφάλματα θα είναι σημαντική. Βασικά, οι προσαρμοστικές εφαρμογές τροποποιούνται πιο εύκολα με αντοχή σε λάθη από ό, τι οι oblivious. Αυτός είναι ο λόγος για τον οποίο

αναμένεται έξαρση της χρησιμοποίησης των προσαρμοζόμενων υλοποιήσεων στο μέλλον.

Η τεχνολογία Network on Chip αναπτύσσεται συνεχώς και μερικές εφαρμογές είναι ήδη έτοιμες για εμπορική χρήση.

Βιβλιογραφία

1. W.J. Dally, B. Towles: *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.

2. M. Dehyadgari, M. Nickray, A. Afzali-kusha, Z. Navabi: *Evaluation of Pseudo Adaptive XY Routing Using an Object OrientedModel for NOC*. The 17th International Conference on Microelectronics, 13–15 December 2005.

3. C. Bobda, A. Ahmadinia, M. Majer, J. Teich, S. Fekete, J. van der Veen: *DyNoC: A Dynamic Infrastructure for Communication in Dynamically Reconfigurable Devices*. International Conference on Field Programmable Logic and Applications, 24–26 August 2005, pages: 153–158.

4. W.J. Dally, B. Towles: *Route Packets, Not Wires: On-Chip Interconnection Networks*. Proceedings, Design Automation Conference 2001, pages: 684–689.

5. M. Yang, T. Li, Y. Jiang, Y. Yang: Fault-Tolerant Routing Schemes in RDT(2,2,1)/-Based Interconnection Network for Networks-on-Chip Designs. Proceedings, 8th International Symposium on Parallel Architectures, Algorithms and Networks, 7–9 December 2005.

6. K. Goossens, J. Dielissen, A. Radulescu: *Æthereal Network on Chip: Concepts, Architectures and Implementations*. IEEE Design & Test of Computers, 2005, Volume 22, Issue 5, pages: 414–421.

7. J. Nurmi: *Network-on-Chip: A New Paradigm for System-on-Chip Design*. Proceedings 2005 International Symposium on System-on-Chip, 15–17 November 2005, pages: 2–6.

8. K. Kim, S. J. Lee, K. Lee, H.J. Yoo: *An Arbitration Look-Ahead Scheme for Reducing End-to-End Latency in Networks on chip*. IEEE International Symposium on Circuits and Systems, 23–26 May 2005, Volume 3, pages: 2357–2360.

9. N. Bansal, A. Blum, S. Chawla, A. Meyerson: *Online Oblivious Routing*. Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures, 2003, pages: 44–49.

10. T.A. Bartic, J.-Y.Mignolet, V. Nollet, T.Marescaux, D. Verkest, S. Vernalde, R. Lauwereins: *Topology adaptive network-on-chip design and implementation*. IEE Proceedings – Computers and Digital Techniques, 8 July 2005, Volume 152, Issue 4, pages: 467–472.

11. M. Pirretti, G.M. Link, R.R. Brooks, N. Vijaykrishnan, M. Kandemir, M.J. Irwin: *Fault Tolerant Algorithms for Networks-On-Chip Interconnect*. Proceedings, IEEE Computer society Annual Symposium on VLSI, 19–20 February 2004, pages: 46–51.

12. J. Kim, D. Park, T. Theocharides, N. Vijaykrishnan, C.R. Das: *A Low Latency Router Supporting Adaptivity for On-Chip Interconnects*. Proceedings, 42. Design Automation Conference, 13–17 June 2005, pages: 559–564.

13. A. Adriahantenaina, H. Charlery, A. Greiner, L. Mortiez, C.A. Zeferino: *SPIN: a Scalable, Packet Switched On-chip Micro-network*. Design, Automation and Test in Europe Conference and Exhibition, 2003, p. 70–73.

14. H. Kariniemi, J. Nurmi: *Fault-tolerant XGFT Network-on-Chip for Multiprocessor System-on-Chip Circuits*. International Conference on Field Programmable Logic and Applications, 24–26 August 2005, pages: 203–210.

15. L.M. Ni, Y. Gui, S. Moore: *Performance Evaluation of Switch-Based Wormhole Networks*. IEEE Transactions on Parallel and Distributed Systems, 1997, Volume 8, Issue 5, pages: 462–474.

16. H. Kariniemi, J. Nurmi: *Arbitration and Routing Schemes for On-chip Packet Networks*. Interconnect-Centric Design for Advanced SoC and NoC (toim: J. Nurmi, H. Tenhunen, J. Isoaho & A. Jantsch), Kluwer Academic Publishers, 2004, pages: 253–282.

17. M.Majer, C. Bobda, A. Ahmadinia, J. Teich: *Packet Routing in Dynamically Changing Networks on Chip*. Proceedings, 19th IEEE International Parallel and Distributed Processing Symposium, 4–8 April 2005, page: 154b.

18. J. Hu, R. Marculescu: *DyAD – Smart Routing for Networks-on-Chip*. Proceedings, 41st Design Automation Conference, 2004, pages: 260–263.

19. D. Andreasson, S. Kumar: *Slack-Time Aware Routing in NoC Systems*. IEEE International Symposium on Circuits and Systems, 23–26 May 2005, pages: 2353–2356.

20. U. Feige, P. Raghavan: *Exact Analysis of Hot-Potato Routing*. 33rd Annual Symposiumon Foundations of Computer Science, 24–27 October 1992, pages: 553–562.

21. N. Kavaldjiev, G.J.M. Smit, P.G. Jansen: *A Virtual Channel Router for Onchip Networks*. Proceedings, IEEE International SOC Conference, 12–15 September 2004, pages: 289–293.

22. M. Dall'Osso, G. Biccari, L. Giovannini, D. Bertozzi, L. Benini: *Xpipes: a Latency Insensitive Parameterized Network-on-chip Architecture For Multi-Processor SoCs*. Proceedings of the 21st International Conference on Computer Design, 13–15 October 2003, pages: 536–539.

23. J. Dielissen, A. Radulescu, K. Goossens, E. Rijpkema: *Concepts and Implementation of the Philips Network-on-Chip*. IP-Based SOC Design, Grenoble, France, Nov 2003.

24. M. Alho, J. Nurmi: *Implementation of interface router IP for Proteo network-on-chip*. The 6th IEEE International Workshop on Design and Diagnostics of Electronics Circuits and Systems, Poznan, Poland, 2003.

25. T. Bjerregaard, J. Sparso: A Router Architecture for Connection-Oriented Service Guarantees in the MANGO Clockless Network-on-Chip. Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, 2005, Volume 2, pages: 1226–1231.

26. D. Wiklund, D. Liu: *SoCBUS: Switched Network on Chip for Hard Real Time Embedded Systems*. Proceedings, International Parallel and Distributed Processing Symposium, 22–26 April 2003.

27. Arteris, http://www.arteris.com/

28. STMicroelectronics. http://www.st.com

29. J. Hu, R. Marculescu: *DyAD – Smart Routing for Networks-on-Chip*. Proceedings, 41st Design Automation Conference, 2004, pages: 260–263.

30. Nostrum, http://www.imit.kth.se/info/FOFU/Nostrum/

Παράρτημα

Source files (Κώδικας προσομοίωσης Noc σε C++)

noc addr.h

```
#ifndef NOC_ADDR_DEF
#define NOC_ADDR_DEF

#include <iostream>

/* Struct presenting a NoC address consisting of two coordinates: x and y.
*/

struct noc_addr{
   int x;
   int y;
   inline noc_addr( ):x( 0 ), y( 0 ){ }
   inline noc_addr( int x, int y ):x( x ), y( y ){ }
};

inline bool operator==( noc_addr n1, noc_addr n2 ){
   return ( ( n1.x == n2.x ) && ( n1.y == n2.y ) );
}

inline bool operator!=( noc_addr n1, noc_addr n2 ){
   return ( ( n1.x != n2.x ) || ( n1.y != n2.y ) );
}

inline std::ostream& operator<<( std::ostream& o, noc_addr na ){
   o << "(" << na.x << ", " << na.y << ")";
   return o;
}

#endif
```

Core.h

```
#ifndef CORE_DEF
#define CORE_DEF

#include <vector>
#include <iostream>

#include "noc_addr.h"
#include "Packet.h"

class NetworkIF;

/* The class represents a core in a NoC. It has an address given at the
initialization. The core is connected to a number of network interfaces,
pointers to which are stored in a vector. The core gives an id to Nis
connecting to it (sending packets to it). The number of such NIs is counted.
Core sends packets using send_packet. The number of sent packets is stored
as well as the addresses of the sent packets. Pointers to the received packets
are stored in a vector as well as the pointers to dropped packets.
*/

class Core{
public:
   Core( const noc_addr addr );
   ~Core( );
   void connect( NetworkIF* n );
   void send_packet( noc_addr dest, int live_time );
   void receive_packet( Packet* p );
   inline noc_addr get_addr( ) const { return addr; }
   inline int get_new_id( ) { return ++ni_count; }
   std::vector< Packet* >* get_received_packets( );
   void report( std::ostream& o );
   void report_sent( std::ostream& o );
   void report_received( std::ostream& o );
   void clear_traffic( );
   inline int get_dropped_count( ) const { return dropped.size( ); }
   private:

   // address of the core
   const noc_addr addr;

   // pointers to NIs the core is connected to (core forwards packets to)
   std::vector< NetworkIF* > ni;
```

```
    // number of NIs connected to core (sends packets to core)
    int ni_count;

    // number of packets sent
    int packet_count;

    // addresses of the sent packets
    std::vector< noc_addr > sent_packets;

    // pointers to the received packets
   std::vector< Packet* > received_packets;

    // pointers to the dropped packets
    std::vector< Packet* > dropped;
};

#endif
```

NetworkIF.h

```
#ifndef NETWORK_IF_DEF
#define NETWORK_IF_DEF

#include <iostream>
#include <vector>
#include "Packet.h"

class Core;
class Link;

/* The class represents a network interface (NI) which is connected to a core
in a NoC. The NI forwards the packets it receives from the link(s) to the core.
The NI is also connected to a number of links into which it forwards the
packets it receives from the core. The NI is identified by the address of the
core it is connected to together with an id it gets from the core. This means
that there can be more than one NIs connected to a single core. The NI also
counts the number of links that are connected to it and gives
ids to them.
*/

class NetworkIF{

public:
   NetworkIF( );
   ~NetworkIF( );
   void connect( Core* co);
   void connect( Link* li);
   void receive_packet( Packet* p );
   void report( std::ostream& o );
   inline noc_addr get_route_to( ) const { return route_to; }
   inline int get_new_id( ) { return ++link_count; }
   void clear_traffic( );
   bool is_usable( );

private:

   // pointer to the core the NI is connected to (NI forwards packets to)
   Core* c;

   // address of the core the NI is connected to
   noc_addr route_to;

   // id number given by the core the NI is connected to
```

```
    int id;

    // vector of pointers to the links the NI forwards packets to
    std::vector< Link* > l;

    // number of links connected to NI (sending packets to NI)
    int link_count;

    // number of packets processed
    int processed;
    void send_packet( Packet* p );
};

#endif
```

Link.h

```
#ifndef LINK_DEF
#define LINK_DEF

#include <iostream>
#include <vector>
#include "Packet.h"

class Router;

/* The class represents an unidirectional link in a NoC. It is connected to a
router from which it gets the route to address and id number (there can be
several addresses). Link forwards packets to this router. The link is also
connected to the router it receives packet from. The link has usability status
for indicating if it can be used or not.
*/

class Link{
  public:
    Link( );
    void connect( Router* ro );
    void connect_back( Router* ro );
    void receive_packet( Packet* p );
    void report( std::ostream& o );
    inline const std::vector< noc_addr >& get_route_to( ) const { return
route_to; }
    inline int get_id( ) const { return id; }
    inline int get_processed( ) const { return processed; }
    inline bool is_usable( ) const { return usable; }
    inline void set_usable( bool u ) { usable = u; }
    void clear_traffic( );

  private:
    // pointer to the router the link is connected to (forwards packets to)
    Router* r;

    // pointer to the router the link is connected to (receives packets from)
    Router* r_b;

    // route to addresses of the router the link is connected to
    std::vector< noc_addr > route_to;

    // an id given by the component the link is connected to
    int id;
```

```
    // number of packets processed
    int processed;

    // status of the link
    bool usable;
    void send_packet( Packet* p );
};

#endif
```

Link ni.h

```
#ifndef LINK_NI_DEF
#define LINK_NI_DEF

#include <iostream>
#include "Packet.h"

class NetworkIF;

/* The class represents an unidirectional link in a NoC connected to a NI
(forwarding packets to NI). It gets the route to address and id number from
the NI. Link forwards packets to this NI. The link has usability status for
indicating if it can be used or not.
*/

class Link_ni{

 public:
  Link_ni( );
  void connect( NetworkIF* n );
  void receive_packet( Packet* p );
  void report( std::ostream& o );
  inline noc_addr get_route_to( ) const { return route_to; }
  inline int get_id( ) const { return id; }
  inline int get_processed( ) const { return processed; }
  inline bool is_usable( ) const { return usable; }
  inline void set_usable( bool u ) { usable = u; }
  void clear_traffic( );

 private:
  // pointer to the NI the link is connected to (forwards packets to)
  NetworkIF* ni;

  // route to address of the NI the link is connected to
  noc_addr route_to;

  // an id given by the component the link is connected to
  int id;

  // number of packets processed
  int processed;

  // status of the link
  bool usable;
```

```
    void send_packet( Packet* p );
};

#endif
```

Router.h

```
#ifndef ROUTER_DEF
#define ROUTER_DEF

#include <iostream>
#include <vector>
#include "noc_addr.h"
#include "Packet.h"

class Link;
class Link_ni;

/* The class represents a router in a NoC. The router is connected via links to
network interfaces and other routers. It forwards packets to the Nis if the
packet destination address matches the route to address of a NI. Otherwise the
packet is forwarded to one of the other routers according to the routing
algorithm. The router is identified by the addresses of the cores it is connected
to (via NIs and links) together with ids which it gets from the NIs (via link).
The router also counts the number of links that are connected to it and gives
ids to them.
*/

class Router{
  public:
    Router( );
    ~Router( );
    void connect( Link* li );
    void connect( Link_ni* li );
     void connect_copy( Link* li); // Special connect-function for tree networks
    void receive_packet( Packet* p, Router* last_router );
    void report( std::ostream& o );
    inline const std::vector< noc_addr >& get_route_to( ) const { return
route_to; }
    inline int get_dropped_count( ) const { return dropped.size( ); }
    inline int get_new_id( ) { return ++link_count; }
    inline int get_processed( ) { return processed; }
    void clear_traffic( );

  private:
    // vector of pointers to the links connected to a NI the router

 // forwards packets to
    std::vector< Link_ni* > l_ni;
```

```
    // addresses of the cores the router is connected to (via a link and NI)
    std::vector< noc_addr > route_to;

    // id numbers given by the NIs the router is connected to (via links)
    std::vector< int > id;

    // vector of pointers to the links connected to other routers the router
    // forwards packets to
    std::vector< Link* > l_router; // Vector for child routers
    std::vector< Link* > l_router_a; // Vector for ancestor routers

    // number of links connected to router (sending packets to router)
    int link_count;

    // number of packets processed
    int processed;

    // vector of pointers to the packets dropped by the router
    std::vector< Packet* > dropped;
    void send_packet( Packet* p, Router* last_router );

};

#endif
```

Packet.h

```
#ifndef PACKET_DEF
#define PACKET_DEF

#include <iostream>
#include "noc_addr.h"

/* The class represents a packet in a NoC. It has a source and destination addresses as well as a number given to it by the source. At construction a live time is given for the packet, which is then decreased at each router. The packet counts hops in NIs, links and routers. */

class Packet{

 public:

  Packet( noc_addr dest, noc_addr source, int nr, int live_time );
  inline noc_addr get_dest( ) const { return dest; }
  inline noc_addr get_source( ) const { return source; }
  inline int get_nr( ) const { return nr; }
  inline int get_router_hops( ) const { return router_hop; }
  inline int get_ni_hops( ) const { return ni_hop; }
  inline int get_link_hops( ) const { return link_hop; }
  inline void add_ni_hop( ) { ni_hop++; }
  inline void add_link_hop( ) { link_hop++; }
  inline void add_router_hop( ) { router_hop++; }
  inline int decrease_live_time( ) { return --live_time; }
  void report( std::ostream& o );

 private:
  // source address
  noc_addr source;

  // a number given by the source
  int nr;

  // destination address
  noc_addr dest;

  // number of NI hops
  int ni_hop;

  // number of link hops
  int link_hop;
```

```
    // number of router hops
    int router_hop;

    // number of router hops before the packet should be dropped
    int live_time;
};

#endif
```

Noc mesh.h

```
#ifndef NOC_MESH_DEF
#define NOC_MESH_DEF

#include <iostream>
#include <boost/multi_array.hpp>
#include "noc_addr.h"
#include "Core.h"
#include "NetworkIF.h"
#include "Link.h"
#include "Link_ni.h"
#include "Router.h"

/* The class represents a mesh-shaped Network on Chip (NoC). It gets the
maximum address as a paramter, minimum address is (0, 0).
*/

class Noc_mesh{

  public:
    Noc_mesh( const noc_addr max_addr );
    ~Noc_mesh( );
    void random_link_error( int n );
    void random_traffic( int n );
    void report( std::ostream& o );
    void report_router( std::ostream& o );
    void report_link( std::ostream& o );
    void report_hop_count( std::ostream& o );
    void report_dropped( std::ostream& o );
    int number_of_dropped( );
    double average_of_hop_count( );
    void clear_traffic_and_state( );

  private:

    // the maximum address in the NoC
    const noc_addr max_addr;

    // array [max_addr.x + 1][max_addr.y + 1] of cores
    boost::multi_array< Core*, 2 >* c;

    // array [max_addr.x + 1][max_addr.y + 1] of network interfaces
    boost::multi_array< NetworkIF*, 2 >* n;
```

```
    // array [max_addr.x + 1][max_addr.y + 1] of routers
    boost::multi_array< Router*, 2 >* r;

    // array [max_addr.x + 1][max_addr.y + 1] of links from routers to NIs
    boost::multi_array< Link_ni*, 2 >* l_ni;

    // array [max_addr.x + 1][max_addr.y + 1] of links from NIs to routers
    boost::multi_array< Link*, 2 >* l_r;

    // array [max_addr.x][max_addr.y + 1][2] of horizontal links between
routers,

    // 3rd dimenision: 0 forward direction, 1 reverse
    boost::multi_array< Link*, 3 >* l_h;

    // array [max_addr.x + 1][max_addr.y][2] of verticalal links between
routers,

     // 3rd dimension: 0 forward direction, 1 reverse

    boost::multi_array< Link*, 3 >* l_v;
    void connect( Core* c, NetworkIF* n);
    void connect( NetworkIF* n, Link_ni* l, Router* r );
    void connect( Router* r, Link* l, NetworkIF* n );
    void connect( Router* r1, Router* r2, Link* l12, Link* l21 );

};

#endif
```

Noc tree.h

```
#ifndef NOC_TREE_DEF
#define NOC_TREE_DEF

#include <iostream>
#include <cmath>
#include <boost/multi_array.hpp>
#include "noc_addr.h"
#include "Core.h"
#include "NetworkIF.h"
#include "Link.h"
#include "Link_ni.h"
#include "Router.h"

/* The class represents a mesh-shaped Network on Chip (NoC). It gets the
maximum address as a paramter, minimum address is (0, 0).
*/

class Noc_tree{

 public:
  Noc_tree( const noc_addr max_addr );
  ~Noc_tree();
  void random_link_error( int n );
  void random_traffic( int n );
  void report( std::ostream& o );
  void report_router( std::ostream& o);
  void report_link( std::ostream& o);
  void report_hop_count( std::ostream& o);
  void report_dropped( std::ostream& o);
  int number_of_dropped( );
  double average_of_hop_count( );
  void clear_traffic_and_state( );

 private:
  const noc_addr max_addr;
  std::vector< Core* > c;
  std::vector< NetworkIF* > n;
  std::vector< Link_ni* > l_ni;
  std::vector< Link* > l_r;
  std::vector< Link* > l;
  std::vector< Router* > r;
  std::vector< Link* > l_t;
  void connect( Core* c, NetworkIF* n);
```

```
    void connect( NetworkIF* n, Link_ni* l, Router* r );
    void connect( Router* r, Link* l, NetworkIF* n );
    void connect( Router* r1, Router* r2, Link* l12, Link* l21 );
    void connect2( Router* rh, Router* rl, Link* lhl, Link* llh );

};


#endif
```